```
                                    H 5



    FFFFFFFFFFFFFFF        111         111     XXX       XXX
    FFFFFFFFFFFFFFF        111         111     XXX       XXX
    FFFFFFFFFFFFFFF        111         111     XXX       XXX
    FFF                 111111      111111     XXX       XXX
    FFF                 111111      111111     XXX       XXX
    FFF                 111111      111111     XXX       XXX
    FFF                    111         111        XXX XXX
    FFF                    111         111        XXX XXX
    FFF                    111         111        XXX XXX
    FFFFFFFF.FFF           111         111           XXX
    FFFFFFFFFFFF           111         111           XXX
    FFFFFFFFFFFF           111         111           XXX
    FFF                    111         111        XXX XXX
    FFF                    111         111        XXX XXX
    FFF                    111         111        XXX XXX
    FFF                    111         111     XXX       XXX
    FFF                    111         111     XXX       XXX
    FFF                    111         111     XXX       XXX
    FFF                 111111111   111111111  XXX       XXX
    FFF                 111111111   111111111  XXX       XXX
    FFF                 111111111   111111111  XXX       XXX
```

```
_$25

Symb
----
IOC1
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

```
**FILE**ID**CLENUP

     CCCCCCCC  LL          EEEEEEEEEE  NN      NN  UU      UU  PPPPPPPP
     CCCCCCCC  LL          EEEEEEEEEE  NN      NN  UU      UU  PPPPPPPP
   CC          LL          EE          NN      NN  UU      UU  PP      PP
   CC          LL          EE          NN      NN  UU      UU  PP      PP
   CC          LL          EE          NNNN    NN  UU      UU  PP      PP
   CC          LL          EE          NNNN    NN  UU      UU  PP      PP
   CC          LL          EEEEEEEE    NN  NN  NN  UU      UU  PPPPPPPP
   CC          LL          EEEEEEEE    NN  NN  NN  UU      UU  PPPPPPPP
   CC          LL          EE          NN    NNNN  UU      UU  PP
   CC          LL          EE          NN    NNNN  UU      UU  PP
   CC          LL          EE          NN      NN  UU      UU  PP
   CC          LL          EE          NN      NN  UU      UU  PP            ....
     CCCCCCCC  LLLLLLLLLL  EEEEEEEEEE  NN      NN  UUUUUUUUUU  PP            ....
     CCCCCCCC  LLLLLLLLLL  EEEEEEEEEE  NN      NN  UUUUUUUUUU  PP            ....
                                                                             ....

   LL          IIIIII      SSSSSSSS
   LL          IIIIII      SSSSSSSS
   LL            II      SS
   LL            II      SS
   LL            II      SS
   LL            II      SS
   LL            II        SSSSSS
   LL            II        SSSSSS
   LL            II              SS
   LL            II              SS
   LL            II              SS
   LL            II              SS
   LLLLLLLLLL  IIIIII    SSSSSSSS
   LLLLLLLLLL  IIIIII    SSSSSSSS
```

```
0 MODULE CLENUP (
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-000'
0                       ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  F11ACP Structure Level 2
1 !
1 ! ABSTRACT:
1 !
1 !       This module performs the necessary cleanup after an operation.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  6-Jan-1977  23:53
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-034 CDS0022         Christian D. Saether    30-Aug-1984
1 !               Allow for multi-header directory files.
1 !               Have error cleanup remove possible bias on primary_fcb
1 !               refcnt.
1 !
1 !       V03-033 CDS0021         Christian D. Saether    23-Aug-1984
1 !               Move code that marks FCB stale to a routine in LOCKERS.
```

```
1 !
1 !      V03-032 CDS0020          Christian D. Saether    13-Aug-1984
1 !              Add code to mark primary fcb stale clusterwide.
1 !
1 !      V03-031 CDS0019          Christian D. Saether     7-Aug-1984
1 !              Cleanup potential directory index cache block
1 !              when deleting a file.
1 !
1 !      V03-030 CDS0018          Christian D. Saether     1-Aug-1984
1 !              Modify test for directory fcb.
1 !              Add SET_DIRINDX routine.
1 !              Add NUKE_PRIM_FCB routine.
1 !              Modify ZERO_IDX routine.
1 !
1 !      V03-029 ACG0438          Andrew C. Goldstein,    19-Jul-1984  17:55
1 !              Add cluster-wide special cache interlock logic.
1 !              Condition DELETEACL calls on non-empty ACL.
1 !              Use central dequeue routine.
1 !
1 !      V03-028 CDS0017          Christian D. Saether    25-May-1984
1 !              Call KILL_BUFFERS routine to flush cache in
1 !              certain situations when not in a cluster.
1 !
1 !      V03-027 CDS0016          Christian D. Saether     9-May-1984
1 !              Release allocation lock prior to calling send_symbiont.
1 !
1 !      V03-026 CDS0015          Christian D. Saether     4-May-1984
1 !              No not map notrunc into nowrite.
1 !              Add bugcheck if access lock conversion fails in make_deaccess.
1 !
1 !      V03-025 CDS0014          Christian D. Saether     3-May-1984
1 !              Call CONV_ACCLOCK to remove possible access lock
1 !              when deallocating fcb's.
1 !
1 !      V03-024 CDS0013          Christian D. Saether    19-Apr-1984
1 !              Changes to FCB$W_ACNT handling.
1 !
1 !      V03-023 ACG0415          Andrew C. Goldstein,     5-Apr-1984  21:27
1 !              Interface change to ACL_DELETEACL
1 !
1 !      V03-022 ACG0408          Andrew C. Goldstein,    23-Mar-1984  11:20
1 !              Make rest of global storage based
1 !
1 !      V03-021 CDS0012          Christian D. Saether     9-Mar-1984
1 !              Put in bug trap to catch possible double remque of
1 !              FCB.
1 !
1 !      V03-020 CDS0011          Christian D. Saether    23-Feb-1984
1 !              Use new WRITE_DIRTY routine to replace FLUSH_BUFFERS.
1 !              Remove references to FLUSH_FID.
1 !              Replace FLUSH_FID (0) with KILL_CACHE calls.
1 !
1 !      V03-019 CDS0010          Christian D. Saether    27-Dec-1983
1 !              Use L_NORM linkage.
1 !              Use BIND_COMMON macro to reduce external declarations.
1 !
1 !      V03-018 CDS0009          Christian D. Saether    23-Nov-1983
```

```
1 !             If DIR_FCB is the same as PRIMARY_FCB, do not return
1 !             the FCB until the end of cleanup (as PRIMARY_FCB, not
1 !             DIR_FCB).
1 !             Move cleanup of DIR_FCB until after all i/o is done.
1 !             Remove REMOVE_FCB routine (kernel call not necessary).
1 !
1 !     V03-017 LMP0164         L. Mark Pilant,         10-Oct-1983  15:22
1 !             Delete the in-core ACL if doing an FCB fixup.
1 !
1 !     V03-016 CDS0008         Christian D. Saether     3-Oct-1983
1 !             Handle CURR_LCKINDX in err_cleanup.  Don't read
1 !             headers without appropriate serial locks.
1 !
1 !     V03-015 CDS0007         Christian D. Saether    14-Sep-1983
1 !             Take out deqall hack now that RMS does it's own
1 !             root locks again.
1 !
1 !     V03-014 CDS0006         Christian D. Saether    27-Jul-1983
1 !             Change interface to SEND_SYMBIONT.
1 !
1 !     V03-013 LJK0199         Lawrence J. Kenah       27-Apr-1983
1 !             Do not credit FILCNT when giving back shared window
1 !
1 !     V03-012 CDS0006         Christian D. Saether    28-Apr-1983
1 !             Clear DIR_ENTRY when DIR_FCB is cleared.
1 !
1 !     V03-011 CDS0005         Christian D. Saether    21-Apr-1983
1 !             Change interface to TRUNCATE routine.
1 !
1 !     V03-010 CDS0004         Christian D. Saether    19-Apr-1983
1 !             Bug check on unexpected lock manager errors.
1 !             Clear ACCLKID field in window.
1 !
1 !     V03-009 ACG0323         Andrew C. Goldstein,    12-Apr-1983  14:09
1 !             Add extended file name to back link fixup
1 !
1 !     V03-008 STJ3069         Steven T. Jeffreys,     23-Mar-1983
1 !             Use the ERASE_REQUESTED parameter of RETURN_BLOCKS.
1 !
1 !     V03-007 CDS0003         Christian D. Saether     7-Mar-1983
1 !             Perform a DEQALL if file access lock dequeue fails
1 !             due to sublocks, then redo the file access dequeue.
1 !
1 !     V03-006 LMP0071         L. Mark Pilant,         19-Jan-1983  20:49
1 !             Correct a problem that caused ACL segments to be left laying
1 !             around when a directory FCB was flushed.
1 !
1 !     V03-005 ACG0308         Andrew C. Goldstein,    14-Jan-1983  15:02
1 !             Fix FCB linkage consistency problems
1 !
1 !     V03-004 CDS0002         Christian D. Saether     3-Jan-1983
1 !             Always flush header cache until it is restored for xqp.
1 !
1 !     V03-003 LMP0059         L. Mark Pilant,         21-Dec-1982  12:23
1 !             Always create an FCB when accessing a file header.  This
1 !             eliminates a lot of special case FCB handling.
1 !
```

```
1 !     V03-002 CDS0001         Christian D. Saether    10-Dec-1982
1 !             MAKE_DEACCESS dequeues access lock.
1 !
1 !     V03-001 LMP0036         L. Mark Pilant,         17-Aug-1982  10:45
1 !             If the ACL was built using a dummy FCB, dismantle and
1 !             deallocate the ACL.
1 !
1 !     V02-024 ACG0259         Andrew C. Goldstein,    26-Jan-1982  19:12
1 !             Add mode arg to REMOVE
1 !
1 !     V02-023 ACG0247         Andrew C. Goldstein,    23-Dec-1981  20:26
1 !             Make /NOCACHE flush all caches
1 !
1 !     V02-022 ACG0245         Andrew C. Goldstein,    23-Dec-1981  20:26
1 !             Send spool file to print during cleanup
1 !
1 !     V02-021 ACG0244         Andrew C. Goldstein,    23-Dec-1981  20:14
1 !             Do buffer flush before deallocating control blocks
1 !
1 !     V02-020 LMP0003         L. Mark Pilant, 30-Nov-1981  16:40
1 !             Properly cleanup any cathedral windows.
1 !
1 !     V02-019 ACG0208         Andrew C. Goldstein,    11-Nov-1981  17:51
1 !             Add segmented directory record support
1 !
1 !     V02-018 ACG0168         Andrew C. Goldstein,    7-May-1980  18:22
1 !             Fix last block directory cleanup on delete failure
1 !
1 !     V02-017 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:25
1 !             Previous revision history moved to F11B.REV
1 !**
1
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 REQUIRE 'SRC$:FCPDEF.B32';
1
1
1 FORWARD ROUTINE
1       CLEANUP         : L_NORM,               ! normal cleanup
1       ZERO_WINDOWS    : L_NORM,               ! invalidate all windows of file
1       ZERO_IDX        : L_NORM NOVALUE,       ! initialize directory index
1       ERR_CLEANUP     : L_NORM,               ! cleanup after error
1       FLUSH_FIDCACHE  : L_NORM,               ! clean out the file ID cache
1       MAKE_DEACCESS   : L_NORM,               ! deaccess the file
1       DEL_EXTFCB      : L_NORM,               ! deallocate extension FCB's
1       ZERO_CHANNEL    : L_NORM,               ! zero user channel pointer
1       SET_DIRINDX     : L_JSB_1ARG,           ! test for directory index
1       NUKE_HEAD_FCB   : L_NORM NOVALUE;       ! deallocate primary fcb
```

```
1 GLOBAL ROUTINE CLEANUP : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine performs the cleanup needed after a successfully
1 !       completed file operation.
1 !
1 ! CALLING SEQUENCE:
1 !       CLEANUP ()
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       CLEANUP_FLAGS: indicate specific actions to do
1 !       PRIMARY_FCB: FCB of file
1 !       CURRENT_WINDOW: window of file
1 !       DIR_FCB: FCB of directory
1 !       CURRENT_VCB: VCB of volume in process
1 !       IO_PACKET: I/O packet of request
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       FCB's and windows deleted when appropriate
1 !       header written
1 !       FCB updated
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2       CLUSTER,                        ! are we a cluster
2       QUOTA_CACHE     : REF BBLOCK,   ! address of quota cache
2       FCB             : REF BBLOCK,   ! local FCB pointer
2       VCB             : REF BBLOCK,   ! local VCB pointer
2       RVT             : REF BBLOCK,   ! local RVT pointer
2       UCB             : REF BBLOCK,   ! local UCB pointer
2       HEADER          : REF BBLOCK;   ! file header
2
2 BIND_COMMON;
2
2 DIR_CONTEXT_DEF;
2
2 EXTERNAL
2       CLU$GL_CLUB     : ADDRESSING_MODE (ABSOLUTE);
2
```

```
2 EXTERNAL ROUTINE
2               MAKE_FCB_STALE   : L_NORM NOVALUE, ! mark fcb as stale clusterwide
2               KILL_BUFFERS     : L_NORM NOVALUE, ! invalidate specified buffers
2               KILL_CACHE       : L_NORM NOVALUE, ! invalidate all buffers for ucb
2               WRITE_DIRTY      : L_NORM,         ! write all dirty buffers
2               SWITCH_VOLUME    : L_NORM,         ! switch to desired volume
2               FLUSH_QUO_CACHE  : L_NORM;         ! flush the quota cache
2
2
2 ! ***** Note: The primary header of the current file is not necessarily
2 ! resident at this point.
2 !
2 ! Switch back to the primary context area if necessary (no normal cleanup
2 ! is ever necessary on secondary context).
2 !
2
2 IF .CONTEXT_SAVE NEQ 0
2 THEN
3       BEGIN
3       CH$MOVE (CONTEXT_SIZE, CONTEXT_SAVE, CONTEXT_START);
3       CONTEXT_SAVE = 0;
3       END;
2
2 CLUSTER = 0;
2 IF .BBLOCK [CURRENT_UCB [UCB$L_DEVCHAR2], DEV$V_CLU]
2       AND .CLU$GL_CLUB NEQ 0
2 THEN
2       CLUSTER = 1;
2
2 ! Check the entire volume set to see if the index file or storage map
2 ! on any volume is write accessed. If so, flush the buffer pool of any
2 ! of their blocks, and flush the file ID and extent caches as appropriate.
2 ! Also, if the volume is mounted /NOCACHE, flush the entire buffer cache.
2 !
2
2 RVT = .CURRENT_VCB[VCB$L_RVT];
2 INCR J FROM 1 TO
3       BEGIN
3       IF .RVT EQL .CURRENT_UCB
4       THEN (UCB = .RVT; 1)
3       ELSE .RVT[RVT$B_NVOLS]
3       END
2 DO
3       BEGIN
3       IF .RVT NEQ .CURRENT_UCB
3       THEN UCB = .VECTOR [RVT[RVT$L_UCBLST], .J-1];
3       IF .UCB NEQ 0
3
3       THEN
4           BEGIN
4           VCB = .UCB[UCB$L_VCB];
4
4           IF .J EQL 1
4           THEN
5               BEGIN
5
5 ! If someone has the quota file write accessed (and it is active), flush it
```

```
5 ! from the buffer pool. (Note that the quota file is located on RVN 1.)
5 !
5
5                       QUOTA_CACHE = .VCB[VCB$L_QUOCACHE];
5                       IF .QUOTA_CACHE NEQ 0
5                       THEN
5                           IF TESTBITSC (QUOTA_CACHE[VCA$V_CACHEFLUSH])
5                           THEN
6                               BEGIN
6                               SWITCH_VOLUME (1);
6                               FLUSH_QUO_CACHE (); ! may create modified buffers
5                               END;
4                       END;                    ! of this is RVN 1 (or single volume)
4
4 ! If the volume is marked for dismount or nocache, flush out all the
4 ! caches.
4 !
4
4               IF .BBLOCK [UCB [UCB$L_DEVCHAR], DEV$V_DMT]
4               OR .VCB[VCB$V_NOCACHE]
4               THEN
5                   BEGIN
5                   SWITCH_VOLUME (.J);
5                   WRITE_DIRTY (0);
5                   KILL_CACHE (.UCB); ! we cannot use the block cache after this
4                   END;
3               END;
2       END;
2
2 ! Write modified buffers.  The various cache purges above may have
2 ! created more dirty buffers than we had at the start of this routine.
2 ! No more dirty buffers can be created for the remainder of this request.
2 !
2
2 WRITE_DIRTY (0);
2
2 ! Invalidate any windows on the file, if requested.
2 !
2
2 IF TESTBITSC (CLEANUP_FLAGS[CLF_INVWINDOW])
2 THEN KERNEL_CALL (ZERO_WINDOWS, .PRIMARY_FCB);
2
2 ! If a directory fcb is left lying about with no use, dispose of it.
2 ! If the directory file is write accessed, flush the buffer pool of any
2 ! blocks that might be resident. Also flush the directory index.
2 ! Cleanup of these fcbs is deferred until all possible errors in the
2 ! cleanup procedure (i/o errors) have already had an opportunity to happen.
2 !
2
2 IF (FCB = .DIR_FCB) NEQ 0
2 THEN
3     BEGIN
3     IF .FCB [FCB$W_REFCNT] EQL 0
3     THEN
4         BEGIN
4         IF .FCB NEQ .PRIMARY_FCB
4         THEN
```

```
4               IF NOT SET_DIRINDX (.FCB)
4               THEN
5                   BEGIN
5                   DEL_EXTFCB (.FCB);
5                   NUKE_HEAD_FCB (.FCB);
4                   END;
4
4           END
4
3       ELSE
4           BEGIN
4           IF .FCB [FCB$W_WCNT] NEQ 0
4           THEN
5               BEGIN
5               SWITCH_VOLUME (.FCB [FCB$W_FID_RVN]);
5               IF NOT .CLUSTER
5               THEN
5                   KILL_BUFFERS (1, .FCB [FCB$L_LOCKBASIS]);
5               ZERO_IDX ();
4               END;
3           END;
3
3 ! Guarantee that no further attempts will be made to do any directory
3 ! related cleanup.  This cleanup code was moved beyond the buffer
3 ! cleanup to avoid the same situation, but clearing the cleanup flags
3 ! makes sure.
3 !
3
3       CLEANUP_FLAGS [CLF_SUPERSEDE] = 0;
3       CLEANUP_FLAGS [CLF_REENTER] = 0;
3       CLEANUP_FLAGS [CLF_REMOVE] = 0;
3       DIR_FCB = 0;
3       DIR_ENTRY = 0;
3
2       END;
2
2 IF (FCB = .PRIMARY_FCB) NEQ 0
2 THEN
3       BEGIN
3
3 ! Check if the fcb has been modified and if so, and this is a cluster,
3 ! cause potential fcbs on other nodes to be marked as stale so they
3 ! will know to rebuild their fcb chains from the file header(s).
3 !
3
3       IF .CLEANUP_FLAGS [CLF_MARKFCBSTALE]
3           AND .CLUSTER
3       THEN
3           MAKE_FCB_STALE (.FCB);
3
3 ! If an FCB is left about with no use, dispose of it.
3 ! Check whether it is a directory fcb first.
3 !
3
3       IF .FCB[FCB$W_REFCNT] EQL 0
3       THEN
3           IF NOT SET_DIRINDX (.FCB)
```

```
;   449            1438  3          THEN
;   450            1439  4              BEGIN
;   451            1440  4
;   452            1441  4              DEL_EXTFCB (.FCB);
;   453            1442  4
;   454            1443  4              NUKE_HEAD_FCB (.FCB);
;   455            1444  4
;   456            1445  4              PRIMARY_FCB = 0;
;   457            1446  3              END;
;   458            1447  2      END;
;   459            1448  2
;   460            1449  2  RETURN 1;
;   461            1450  2
;   462            1451  1  END;                                                ! end of routine CLEANUP



                                                                    .TITLE  CLENUP
                                                                    .IDENT  \V04-000\

                                                                    .EXTRN  CLU$GL_CLUB, MAKE_FCB_STALE
                                                                    .EXTRN  KILL_BUFFERS, KILL_CACHE
                                                                    .EXTRN  WRITE_DIRTY, SWITCH_VOLUME
                                                                    .EXTRN  FLUSH_QUO_CACHE

                                                                    .PSECT  $CODE$,NOWRT,2

                                      0BFC 00000                    .ENTRY  CLEANUP, Save R2,R3,R4,R5,R6,R7,R8,R9,R11   ; 1210
                      5B      0000G  CF  9E 00002                   MOVAB   SWITCH_VOLUME, R11                           ;
                      58      00DC   CA  9E 00007                   MOVAB   220(BASE), R8                                ; 1258
                                36   AA  D5 0000C                   TSTL    54(BASE)                                     ; 1283
                                     08  13 0000F                   BEQL    1$                                           ;
         6A          36  AA          36  28 00011                   MOVC3   #54, 54(BASE), (BASE)                        ; 1286
                                36   AA  D4 00016                   CLRL    54(BASE)                                     ; 1287
                                     59  D4 00019 1$:               CLRL    CLUSTER                                      ; 1290
                      50        94   AA  D0 0001B                   MOVL    -108(BASE), R0                               ; 1291
                      0B        3C   A0  E9 0001F                   BLBC    60(R0), 2$                                   ;
                           00000000G 9F  D5 00023                   TSTL    @#CLU$GL_CLUB                                ; 1292
                                     03  13 00029                   BEQL    2$                                           ;
                      59             01  D0 0002B                   MOVL    #1, CLUSTER                                  ; 1294
                      50        98   AA  D0 0002E 2$:               MOVL    -104(BASE), R0                               ; 1302
                      52        20   A0  D0 00032                   MOVL    32(R0), RVT                                  ;
                 94   AA             52  D1 00036                   CMPL    RVT, -108(BASE)                              ; 1305
                                     08  12 0003A                   BNEQ    3$                                           ;
                      54             52  D0 0003C                   MOVL    RVT, UCB                                     ; 1306
                      57             01  D0 0003F                   MOVL    #1, R7                                       ;
                                     04  11 00042                   BRB     4$                                           ;
                      57        0B   A2  9A 00044 3$:               MOVZBL  11(RVT), R7                                  ; 1307
                                     53  D4 00048 4$:               CLRL    J                                            ; 1303
                                     4A  11 0004A                   BRB     9$                                           ;
                 94   AA             52  D1 0004C 5$:               CMPL    RVT, -108(BASE)                              ; 1311
                                     05  13 00050                   BEQL    6$                                           ;
                      54        40 A243  D0 00052                   MOVL    64(RVT)[J], UCB                              ; 1312
                                     54  D5 00057 6$:               TSTL    UCB                                          ; 1313
                                     3B  13 00059                   BEQL    9$                                           ;
                      55        34   A4  D0 0005B                   MOVL    52(UCB), VCB                                 ; 1317
                      01             53  D1 0005F                   CMPL    J, #1                                        ; 1319
                                     15  12 00062                   BNEQ    7$                                           ;
```

```
                                                  D 12
CLENUP                                      16-Sep-1984 00:02:25    VAX-11 Bliss-32 V4.0-742                Page  10
V04-000                                     14-Sep-1984 12:30:12    DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1        (2)

                          56      5C   A5  D0 00064         MOVL    92(VCB), QUOTA_CACHE                      : 1327
                                       0F  13 00068         BEQL    7$                                        : 1328
          0A        0B    A6           01  E5 0006A         BBCC    #1, 11(QUOTA_CACHE), 7$                   : 1330
                                       01  DD 0006F         PUSHL   #1                                        : 1333
                          6B           01  FB 00071         CALLS   #1, SWITCH_VOLUME
                 0000G    CF           00  FB 00074         CALLS   #0, FLUSH_QUO_CACHE                       : 1334
          05        3A    A4           05  E0 00079 7$:     BBS     #5, 58(UCB), 8$                           : 1342
          13        53    A5           01  E1 0007E         BBC     #1, 83(VCB), 9$                           : 1343
                                       53  DD 00083 8$:     PUSHL   J                                         : 1346
                          6B           01  FB 00085         CALLS   #1, SWITCH_VOLUME
                                       7E  D4 00088         CLRL    -(SP)                                     : 1347
                 0000G    CF           01  FB 0008A         CALLS   #1, WRITE_DIRTY
                                       54  DD 0008F         PUSHL   UCB                                       : 1348
                 0000G    CF           01  FB 00091         CALLS   #1, KILL_CACHE
          B2              53           57  F3 00096 9$:     AOBLEQ  R7, J, 5$                                 : 1303
                                       7E  D4 0009A         CLRL    -(SP)                                     : 1358
                 0000G    CF           01  FB 0009C         CALLS   #1, WRITE_DIRTY
          08              6A           04  E5 000A1         BBCC    #4, (BASE), 10$                           : 1363
                                  08   AA  DD 000A5         PUSHL   8(BASE)                                   : 1364
                 0000V    CF           01  FB 000A8         CALLS   #1, ZERO_WINDOWS
                          53    00D0   CA  D0 000AD 10$:    MOVL    208(BASE), FCB                            : 1373
                                       50  13 000B2         BEQL    14$
                                  18   A3  B5 000B4         TSTW    24(FCB)                                   : 1376
                                       1F  12 000B7         BNEQ    11$
                    08    AA           53  D1 000B9         CMPL    FCB, 8(BASE)                              : 1379
                                       37  13 000BD         BEQL    13$
                          50           53  D0 000BF         MOVL    FCB, R0                                   : 1381
                                    0000V  30 000C2         BSBW    SET_DIRINDX
                          2E           50  E8 000C5         BLBS    R0, 13$
                                       53  DD 000C8         PUSHL   FCB                                       : 1384
                 0000V    CF           01  FB 000CA         CALLS   #1, DEL_EXTFCB
                                       53  DD 000CF         PUSHL   FCB                                       : 1385
                 0000V    CF           01  FB 000D1         CALLS   #1, NUKE_HEAD_FCB
                                       1E  11 000D6         BRB     13$                                       : 1376
                                  1C   A3  B5 000D8 11$:    TSTW    28(FCB)                                   : 1392
                                       19  13 000DB         BEQL    13$
                          7E      28   A3  3C 000DD         MOVZWL  40(FCB), -(SP)                            : 1395
                          6B           01  FB 000E1         CALLS   #1, SWITCH_VOLUME
                          0A           59  E8 000E4         BLBS    CLUSTER, 12$                              : 1396
                                  4C   A3  DD 000E7         PUSHL   76(FCB)                                   : 1398
                                       01  DD 000EA         PUSHL   #1
                 0000G    CF           02  FB 000EC         CALLS   #2, KILL_BUFFERS
                 0000V    CF           00  FB 000F1 12$:    CALLS   #0, ZERO_IDX                              : 1399
                          6A 00C00020  8F  CA 000F6 13$:    BICL2   #12582944, (BASE)                         : 1411
                                00D0   CA  D4 000FD         CLRL    208(BASE)                                 : 1412
                                  08   A8  D4 00101         CLRL    8(R8)                                     : 1413
                          53      08   AA  D0 00104 14$:    MOVL    8(BASE), FCB                              : 1417
                                       2D  13 00108         BEQL    16$
          0A              6A           0E  E1 0010A         BBC     #14, (BASE), 15$                          : 1426
                          07           59  E9 0010E         BLBC    CLUSTER, 15$                              : 1427
                                       53  DD 00111         PUSHL   FCB                                       : 1429
                 0000G    CF           01  FB 00113         CALLS   #1, MAKE_FCB_STALE
                                  18   A3  B5 00118 15$:    TSTW    24(FCB)                                   : 1435
                                       1A  12 0011B         BNEQ    16$
                          50           53  D0 0011D         MOVL    FCB, R0                                   : 1437
                                    0000V  30 00120         BSBW    SET_DIRINDX
                          11           50  E8 00123         BLBS    R0, 16$                                   :
```

```
                                      53  DD 00126           PUSHL   FCB                                                  ; 1441
                   0000V  CF          01  FB 00128           CALLS   #1, DEL_EXTFCB                                       ;
                                      53  DD 0012D           PUSHL   FCB                                                  ; 1443
                   0000V  CF          01  FB 0012F           CALLS   #1, NUKE_HEAD_FCB                                    ;
                                 08   AA  D4 00134           CLRL    8(BASE)                                              ; 1445
                          50          01  D0 00137 16$:      MOVL    #1, R0                                               ; 1449
                                          04 0013A           RET                                                          ; 1451

; Routine Size:  315 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 GLOBAL ROUTINE ZERO_WINDOWS (FCB) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine invalidates all windows currently in use on the
1 !       indicated FCB. This routine must be executed in kernel mode.
1 !
1 ! CALLING SEQUENCE:
1 !       ZERO_WINDOWS (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of FCB
1 !
1 ! IMPLICIT INPUTS:
1 !       CURRENT_WINDOW: address of caller's window, if any
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       all windows marked empty, caller's turned
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       FCB             : REF BBLOCK;
2
2 LOCAL
2       P               : REF BBLOCK,   ! window pointer
2       DUMMY,                          ! dummy sturage for REMQUE return
2       WINDOW_SEGMENT  : REF BBLOCK,   ! pointer to window segment
2       NEXT_SEGMENT    : REF BBLOCK;   ! pointer to window after next one
2
2 BASE_REGISTER;
2
2 EXTERNAL ROUTINE
2       DEALLOCATE      : L_NORM;       ! deallocate dynamic memory
2
2 ! Loop through the window list off the FCB, zeroing all the retrieval pointer
2 ! counts. Then turn the user's window to VBN 1 if it exists.
2 !
2
2 P = .FCB[FCB$L_WLFL];
2
2 UNTIL .P EQL FCB[FCB$L_WLFL] DO
3     BEGIN
3     P[WCB$W_NMAP] = 0;
```

```
;  521      1509 3        WINDOW_SEGMENT = .P[WCB$L_LINK];
;  522      1510 3        UNTIL .WINDOW_SEGMENT EQL 0
;  523      1511 3        DO
;  524      1512 4            BEGIN
;  525      1513 4            NEXT_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
;  526      1514 4            REMQUE (.WINDOW_SEGMENT, DUMMY);
;  527      1515 4            DEALLOCATE (.WINDOW_SEGMENT);
;  528      1516 4            WINDOW_SEGMENT = .NEXT_SEGMENT;
;  529      1517 3            END;
;  530      1518 3        P[WCB$L_LINK] = 0;
;  531      1519 3        P[WCB$V_COMPLETE] = 0;
;  532      1520 3        P = .P[WCB$L_WLFL];
;  533      1521 2        END;
;  534      1522 2
;  535      1523 2 ! ***** Note: When handling of window misses goes into its final form,
;  536      1524 2 ! this routine must also scan the I/O queue on the UCB and look for I/O
;  537      1525 2 ! into the blocks just deallocated. All such requests must be yanked out
;  538      1526 2 ! of the queue and routed to the ACP for error processing.
;  539      1527 2
;  540      1528 2 RETURN 1;
;  541      1529 2
;  542      1530 1 END;                                         ! end of routine ZERO_WINDOWS


                                                             .EXTRN  DEALLOCATE

                                          003C 00000         .ENTRY  ZERO_WINDOWS, Save R2,R3,R4,R5           ; 1452
                         50      04    AC   D0 00002         MOVL    FCB, R0                                  ; 1504
                         52      10    A0   D0 00006         MOVL    16(R0), P                                ;
          50      04     AC            10   C1 0000A 1$:     ADDL3   #16, FCB, R0                             ; 1506
                         50            52   D1 0000F         CMPL    P, R0                                    ;
                                       28   13 00012         BEQL    4$                                       ;
                                 16    A2   B4 00014         CLRW    22(P)                                    ; 1508
                         53      20    A2   D0 00017         MOVL    32(P), WINDOW_SEGMENT                    ; 1509
                                       13   13 0001B 2$:     BEQL    3$                                       ; 1510
                         54      20    A3   D0 0001D         MOVL    32(WINDOW_SEGMENT), NEXT_SEGMENT         ; 1513
                         55            63   0F 00021         REMQUE  (WINDOW_SEGMENT), DUMMY                  ; 1514
                                       53   DD 00024         PUSHL   WINDOW_SEGMENT                           ; 1515
                 0000G   CF            01   FB 00026         CALLS   #1, DEALLOCATE                           ;
                         53            54   D0 0002B         MOVL    NEXT_SEGMENT, WINDOW_SEGMENT             ; 1516
                                       EB   11 0002E         BRB     2$                                       ; 1510
                                 20    A2   D4 00030 3$:     CLRL    32(P)                                    ; 1518
                    0B   A2            20   8A 00033         BICB2   #32, 11(P)                               ; 1519
                         52            62   D0 00037         MOVL    (P), P                                   ; 1520
                                       CE   11 0003A         BRB     1$                                       ; 1506
                         50            01   D0 0003C 4$:     MOVL    #1, R0                                   ; 1528
                                            04 0003F         RET                                              ; 1530

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 013B
;  544          1531  1 GLOBAL ROUTINE ZERO_IDX : L_NORM NOVALUE =
;  545          1532  1
;  546          1533  1 !++
;  547          1534  1 !
;  548          1535  1 ! FUNCTIONAL DESCRIPTION:
;  549          1536  1 !
;  550          1537  1 !       This routine initializes the index in a directory FCB to an unknown
;  551          1538  1 !       state. It will be rebuilt with the next several lookups.
;  552          1539  1 !       It also bumps the sequence count to indicate a change in contents.
;  553          1540  1 !
;  554          1541  1 !
;  555          1542  1 ! CALLING SEQUENCE:
;  556          1543  1 !       ZERO_IDX ()
;  557          1544  1 !
;  558          1545  1 ! INPUT PARAMETERS:
;  559          1546  1 !       NONE
;  560          1547  1 !
;  561          1548  1 ! IMPLICIT INPUTS:
;  562          1549  1 !       DIR_FCB: directory FCB to init
;  563          1550  1 !
;  564          1551  1 ! OUTPUT PARAMETERS:
;  565          1552  1 !       NONE
;  566          1553  1 !
;  567          1554  1 ! IMPLICIT OUTPUTS:
;  568          1555  1 !       NONE
;  569          1556  1 !
;  570          1557  1 ! ROUTINE VALUE:
;  571          1558  1 !       1
;  572          1559  1 !
;  573          1560  1 ! SIDE EFFECTS:
;  574          1561  1 !       directory index zeroed
;  575          1562  1 !
;  576          1563  1 !--
;  577          1564  1
;  578          1565  2 BEGIN
;  579          1566  2
;  580          1567  2 BIND_COMMON;
;  581          1568  2
;  582          1569  2 LOCAL
;  583          1570  2         DIRINDX : REF BBLOCK FIELD (DIRC);
;  584          1571  2
;  585          1572  2 DIR_FCB[FCB$W_DIRSEQ] = .DIR_FCB[FCB$W_DIRSEQ] + 1;
;  586          1573  2
;  587          1574  2 IF (DIRINDX = .DIR_FCB [FCB$L_DIRINDX]) NEQ 0
;  588          1575  2 THEN
;  589          1576  2     DIRINDX [DIRC$W_INUSE] = 0;
;  590          1577  2
;  591          1578  1 END;                                    ! end of routine ZERO_IDX



                                          0000 00000          .ENTRY   ZERO_IDX, Save nothing                  ; 1531
                      50      00D0   CA   D0 00002            MOVL     208(BASE), R0                           ; 1572
                                42   A0   B6 00007            INCW     66(R0)                                  ;
                      50      00D0   CA   D0 0000A            MOVL     208(BASE), R0                           ; 1574
```

```
                        50      00B0   C0  D0 0000F          MOVL    176(R0), DIRINDX                                    ;
                                       02  13 00014          BEQL    1$                                                  ;
                                       60  B4 00016          CLRW    (DIRINDX)                                           ; 1576
                                           04 00018 1$:      RET                                                         ; 1578

; Routine Size:  25 bytes,    Routine Base:  $CODE$ + 017B
```

```
1 GLOBAL ROUTINE ERR_CLEANUP : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine performs the cleanup needed after a file
1 !       operation that has terminated in an error.
1 !
1 ! CALLING SEQUENCE:
1 !       ERR_CLEANUP ()
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       CLEANUP_FLAGS: indicate specific actions to do
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       file deaccessed if necessary
1 !       channel window pointer cleared
1 !
1 !--
1
2 BEGIN
2
2 BIND_COMMON;
2
2 DIR_CONTEXT_DEF;
2
2 EXTERNAL ROUTINE
2         REBLD_PRIM_FCB   : L_NORM NOVALUE, ! rebuild primary fcb from header
2         BUILD_EXT_FCBS   : L_NORM NOVALUE, ! build extension fcb chain
2         ALLOCATION_UNLOCK : L_NORM NOVALUE, ! release allocation lock
2         KILL_DINDX       : L_NORM NOVALUE, ! release directory index block
2         PMS_END_SUB      : L_NORM,         ! end metering of current subfunction
2         CLOSE_FILE       : L_NORM,         ! close internal file
2         DEACC_QFILE      : L_NORM,         ! deaccess the quota file
2         DEALLOCATE       : L_NORM,         ! deallocate dynamic memory
2         SEND_SYMBIONT    : L_NORM ADDRESSING_MODE (GENERAL),
2                                            ! send file to job controller
2         SWITCH_VOLUME    : L_NORM,         ! switch to desired volume
2         RESTORE_DIR      : L_NORM,         ! restore directory context
2         DIR_SCAN         : L_NORM,         ! scan directory file
2         MAKE_ENTRY       : L_NORM,         ! create new directory entry
2         REMOVE           : L_NORM,         ! remove a directory entry
2         READ_BLOCK       : L_NORM,         ! read a disk block
2         MARK_DIRTY       : L_NORM,         ! mark disk block for write back
```

```
2               WRITE_BLOCK       : L_NORM,         ! write a disk block
2               DELETE_FILE       : L_NORM,         ! delete a file
2               DELETE_FID        : L_NORM,         ! delete a file number
2               RETURN_BLOCKS     : L_NORM,         ! return blocks to storage map
2               TRUNCATE          : L_NORM,         ! file truncate routine
2               INVALIDATE        : L_NORM,         ! invalidate a buffer
2               READ_HEADER       : L_NORM,         ! read file header
2               CHECKSUM          : L_NORM,         ! checksum file header
2               REMAP_FILE        : L_NORM;         ! rebuild the windows for a file
2
2
2  ! If a subfunction was being executed, turn off metering now.
2  !
2
2  IF .PMS_SUB_NEST NEQ 0
2  THEN
3      BEGIN
3      PMS_SUB_NEST = 1;
3      PMS_END_SUB ();
3      END;
2
2  ! We repeat the entire procedure twice if a secondary file operation was
2  ! in progress (indicated by non-zero saved context).
2  !
2
2  WHILE 1 DO
3  BEGIN
3
3  ! Locals are declared here to prevent their scope from extending around the
3  ! entire main loop and raising havoc with register assignment.
3  !
3
3  LOCAL
3               NAME_DESC         : BBLOCK [FND_LENGTH], ! file name descriptor block
3               HEADER            : REF BBLOCK,     ! address of file header
3               IDENT_AREA        : REF BBLOCK,     ! ident area of file header
3               FCB               : REF BBLOCK,     ! FCB pointer
3               WINDOW_SEGMENT    : REF BBLOCK,     ! address of the next window segment
3               NEXT_SEGMENT      : REF BBLOCK,     ! address of one beyond the next window
3               RECADDR           : REF BBLOCK,     ! address of directory record
3               DIR_FLAGS         : BITVECTOR [32], ! directory cleanup flags
3               UNREC_LOCAL,                        ! local copy of UNREC_COUNT
3               FID_LOCAL,                          ! local copy of NEW_FID
3               T1,                                 ! random temps
3               T2,
3               T3;
3
3  ! Show that cleanup is in progress.
3  !
3
3  CLEANUP_FLAGS[CLF_CLEANUP] = 1;
3
3  ! If the ref count on the primary fcb was biased in fid_to_spec, remove
3  ! the bias.
3  !
3
3  IF TESTBITSC (CLEANUP_FLAGS [CLF_PFCB_REF_UP])
```

```
3 THEN
3     PRIMARY_FCB [FCB$W_REFCNT] = .PRIMARY_FCB [FCB$W_REFCNT] - 1;
3
3 ! If an internal file is open, close it first.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_CLOSEFILE])
3 THEN CLOSE_FILE (.CURRENT_WINDOW);
3
3 ! Invalidate the file ID cache, if necessary.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_FLUSHFID])
3 THEN KERNEL_CALL (FLUSH_FIDCACHE);
3
3 ! Deaccess the quota file, if we were in the final stages of a quota file
3 ! enable.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_DEACCQFILE])
3 THEN KERNEL_CALL (DEACC_QFILE);
3
3 ! If there is a file header resident, it probably needs to be checksummed.
3 !
3
3 IF .FILE_HEADER NEQ 0
3 THEN CHECKSUM (.FILE_HEADER);
3
3 ! Clean out the window pointer in the user's channel if necessary.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_ZCHANNEL])
3 THEN KERNEL_CALL (ZERO_CHANNEL);
3
3 ! If there are unrecorded blocks allocated from the storage map, return them.
3 !
3
3 IF (UNREC_LOCAL = .UNREC_COUNT) NEQ 0
3 THEN
4     BEGIN
4     UNREC_COUNT = 0;
4     SWITCH_VOLUME (.UNREC_RVN);
4     RETURN_BLOCKS (.UNREC_LBN, .UNREC_LOCAL, DO_NOT_ERASE);
3     END;
3
3 ! If there is a dangling file ID (from a partial create or header extension),
3 ! dispose of it.
3 !
3
3 IF (FID_LOCAL = .NEW_FID) NEQ 0
3 THEN
4     BEGIN
4     NEW_FID = 0;
4     SWITCH_VOLUME (.NEW_FID_RVN);
4     DELETE_FID (.FID_LOCAL);
3     END;
3
```

```
3 ! Get back the primary file header of the file in process.
3 !
3
3 HEADER = 0;
3 IF .FILE_HEADER NEQ 0
3 THEN
4     BEGIN
4     FILE_HEADER = 0;
4     IF (CURR_LCKINDX = .PRIM_LCKINDX) NEQ 0
4     THEN
5         HEADER = READ_HEADER ((IF .CURRENT_FIB NEQ 0
5                               THEN CURRENT_FIB[FIB$W_FID]
4                               ELSE 0),
4                              .PRIMARY_FCB);
3     END;
3
3 ! Send the file to the job controller if it is to be spooled.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_DOSPOOL])
3 THEN
4     BEGIN
4
4 ! Make sure the allocation lock is released before sending it
4 ! to the symbiont to avoid potential deadlock with the symbiont.
4 !
4
4     ALLOCATION_UNLOCK ();
4     SEND_SYMBIONT (.HEADER, .PRIMARY_FCB);
3     END;
3
3 ! Deaccess the file if requested.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_DEACCESS])
3 THEN KERNEL_CALL (MAKE_DEACCESS);
3
3 ! Deallocate the window block if called for.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_DELWINDOW])
3 THEN
3     IF .CURRENT_WINDOW NEQ 0
3     THEN
4         BEGIN
4         WINDOW_SEGMENT = .CURRENT_WINDOW;
4         DO
5             BEGIN
5             NEXT_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
5             KERNEL_CALL (DEALLOCATE, .WINDOW_SEGMENT);
5             WINDOW_SEGMENT = .NEXT_SEGMENT;
5             END
4         UNTIL .WINDOW_SEGMENT EQL 0;
4         CURRENT_WINDOW = 0;
3         END;
3
3 ! Fix the file header back link, if it was modified.
```

```
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_FIXLINK])
3 THEN IF .HEADER NEQ 0
3 THEN
4     BEGIN
4     CH$MOVE (FID$C_LENGTH, PREV_LINK, HEADER[FH2$W_BACKLINK]);
4     IDENT_AREA = .HEADER + .HEADER[FH2$B_IDOFFSET]*2;
4     CH$MOVE (MINU (FILENAME_LENGTH, FI2$S_FILENAME), PREV_INAME,
4             IDENT_AREA[FI2$T_FILENAME]);
4     CH$MOVE (MINU (FILENAME_LENGTH-FI2$S_FILENAME, FI2$S_FILENAMEXT),
4             PREV_INAME+FI2$S_FILENAME,
4             IDENT_AREA[FI2$T_FILENAMEXT]);
4     CHECKSUM (.HEADER);
4     MARK_DIRTY (.HEADER);
3     END;
3
3 ! If a file deletion is called for, do it.  This is either a create that
3 ! failed later on, or a real delete.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_DELFILE])
3 THEN IF .HEADER NEQ 0
3 THEN
4     BEGIN
4     IF .PRIMARY_FCB NEQ 0
4     THEN
4         IF .PRIMARY_FCB [FCB$L_DIRINDX] NEQ 0
4         THEN
4             KILL_DINDX (.PRIMARY_FCB);
4
4     CLEANUP_FLAGS[CLF_TRUNCATE] = 0;    ! no truncate necessary after a delete
4     DELETE_FILE (.CURRENT_FIB, .HEADER);
3     END;
3
3 ! If an extend operation failed, truncate the file.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_TRUNCATE])
3 THEN IF .HEADER NEQ 0
3 THEN
4     BEGIN
4     T1 = .CURRENT_FIB[FIB$L_EXSZ];      ! save the data returned by EXTEND
4     T2 = .CURRENT_FIB[FIB$L_EXVBN];     ! so it won't be smashed by TRUNCATE
4     T3 = .USER_STATUS[1];
4     CURRENT_FIB[FIB$L_EXSZ] = 0;
4     TRUNCATE (.CURRENT_FIB, .HEADER, .T2);
4     HEADER = .FILE_HEADER;                      ! follow buffer shuffling
4     CURRENT_FIB[FIB$L_EXSZ] = .T1;
4     CURRENT_FIB[FIB$L_EXVBN] = .T2;
4     USER_STATUS[1] = .T3;
4     CLEANUP_FLAGS[CLF_INVWINDOW] = 0;   ! windows were never extended, so no need
4     CHECKSUM (.HEADER);
3     END;
3
3 ! Various errors leave the file control block screwed up. If needed,
3 ! rebuild it and its extensions from scratch.
```

```
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_FIXFCB])
3 AND .HEADER NEQ 0
3 THEN
4     BEGIN
4
4     REBLD_PRIM_FCB (.PRIMARY_FCB, .HEADER);
4
4     BUILD_EXT_FCBS (.HEADER);
4
3     END;
3
3 ! Cleanup any cathedral windows which have broken.
3 !
3
3 IF TESTBITSC (CLEANUP_FLAGS[CLF_REMAP]) THEN REMAP_FILE ();
3
3 ! Do directory operation cleanups. We could have entered a new file, removed
3 ! an old one, or both, or done a supersede. A supersede is a replacement of
3 ! the FID for the same name, type, and version.
3 !
3
3 DIR_FLAGS = .CLEANUP_FLAGS;
3 CLEANUP_FLAGS[CLF_SUPERSEDE] = 0;
3 CLEANUP_FLAGS[CLF_REENTER] = 0;
3 CLEANUP_FLAGS[CLF_REMOVE] = 0;
3
3 IF .DIR_FLAGS[CLF_SUPERSEDE]
3 OR .DIR_FLAGS[CLF_REENTER]
3 OR .DIR_FLAGS[CLF_REMOVE]
3 THEN
4     BEGIN
4     SWITCH_VOLUME (.CURRENT_FIB[FIB$W_DID_RVN]);
4
4 ! Buffer pool thrashing may have kicked out the directory block we need.
4 ! re-read it and recompute the buffer pointers.
4 !
4
4     IF .DIR_ENTRY NEQ 0
4     THEN RESTORE_DIR (DIR_CONTEXT);
4
4 ! If a directory entry needs to be removed, do so. Pointers are all set
4 ! up for the REMOVE routine.
4 !
4
4     IF .DIR_FLAGS[CLF_REMOVE]
4     THEN REMOVE (0);
4
4 ! If a directory entry needs to be re-entered, do so. If the entry was
4 ! removed theough an auto-purge, we need to rescan to the point of
4 ! removal because a directory shuffle may have invalidated the
4 ! pointers. Construct a name descriptor from the saved name and version
4 ! and call the enter routine.
4 !
4
4     IF .DIR_FLAGS[CLF_REENTER]
```

```
4       THEN
5           BEGIN
5           CH$FILL (0, FND_LENGTH, NAME_DESC);
5           NAME_DESC[FND_COUNT] = .PREV_NAME[0];
5           NAME_DESC[FND_STRING] = PREV_NAME[1];
5           NAME_DESC[FND_VERSION] = .PREV_VERSION;
5           IF .DIR_FLAGS[CLF_SUPERSEDE]
5           THEN
6               BEGIN
6               LAST_ENTRY[0] = 0;
6               DIR_SCAN (NAME_DESC, 0, 0, 0, 0, 0, -1);
6               CH$MOVE (FID$C_LENGTH, SUPER_FID, CURRENT_FIB[FIB$W_FID]);
5               END;
5           MAKE_ENTRY (NAME_DESC, .CURRENT_FIB);
5           CLEANUP_FLAGS[CLF_REMOVE] = 0;
5           WRITE_BLOCK (.DIR_BUFFER);
4           END;
4
4    ! A supersede cleanup consists simply of replacing the superseded file ID
4    ! in the directory record. Note that the supersede bit could also be set
4    ! by a create/auto-purge, which also sets the remove and enter bits, and
4    ! is handled above.
4    !
4
4       IF .DIR_FLAGS[CLF_SUPERSEDE]
4       AND NOT .DIR_FLAGS[CLF_REENTER]
4       AND NOT .DIR_FLAGS[CLF_REMOVE]
4       THEN
5           BEGIN
5           DIR_VERSION[DIR$W_VERSION] = .PREV_VERSION;
5           CH$MOVE (FIB$S_FID, SUPER_FID, DIR_VERSION[DIR$W_FID]);
5           MARK_DIRTY (.DIR_BUFFER);
5           END
5
5
3       END;                                     ! end of directory cleanup processing
3
3    ! Copy the saved context, if any back into the primary context and repeat
3    ! the cleanup.
3    !
3
3   IF .CONTEXT_SAVE EQL 0 THEN EXITLOOP;
3   CH$MOVE (CONTEXT_SIZE, CONTEXT_SAVE, CONTEXT_START);
3   CONTEXT_SAVE = 0;
3
2   END;                                         ! end of major loop
2
2   RETURN 1;
2
1   END;                                         ! end of routine ERR_CLEANUP


                                                             .EXTRN  REBLD_PRIM_FCB, BUILD_EXT_FCBS
                                                             .EXTRN  ALLOCATION_UNLOCK
                                                             .EXTRN  KILL_DINDX, PMS_END_SUB
                                                             .EXTRN  CLOSE_FILE, DEACC_QFILE
```

```
CLENUP                                  D 13
V04-000                                 16-Sep-1984 00:02:25    VAX-11 Bliss-32 V4.0-742               Page  23
                                        14-Sep-1984 12:30:12    DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1     (5)

                                   .EXTRN  SEND_SYMBIONT, RESTORE_DIR
                                   .EXTRN  DIR_SCAN, MAKE_ENTRY
                                   .EXTRN  REMOVE, READ_BLOCK
                                   .EXTRN  MARK_DIRTY, WRITE_BLOCK
                                   .EXTRN  DELETE_FILE, DELETE_FID
                                   .EXTRN  RETURN_BLOCKS, TRUNCATE
                                   .EXTRN  INVALIDATE, READ_HEADER
                                   .EXTRN  CHECKSUM, REMAP_FILE

                    0BFC 00000     .ENTRY  ERR_CLEANUP, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 1579
                                           R11
          5E       10 C2 00002     SUBL2   #16, SP
                80 AA 9F 00005     PUSHAB  -128(BASE)                                    ; 1612
                08 AA 9F 00008     PUSHAB  8(BASE)
          59    10 AA 9E 0000B     MOVAB   16(BASE), R9
          57  00DC CA 9E 0000F     MOVAB   220(BASE), R7
          5B  01A8 CA 9E 00014     MOVAB   424(BASE), R11
              0908 CA D5 00019     TSTL    2312(BASE)                                    ; 1650
                   0A 13 0001D     BEQL    1$
     0908 CA       01 D0 0001F     MOVL    #1, 2312(BASE)                                ; 1653
    0000G CF       00 FB 00024     CALLS   #0, PMS_END_SUB                               ; 1654
       01 AA       02 88 00029 1$: BISB2   #2, 1(BASE)                                   ; 1686
07        6A       0F E5 0002D     BBCC    #15, (BASE), 2$                               ; 1692
          50    00 BE D0 00031     MOVL    @0(SP), R0                                    ; 1694
                18 A0 B7 00035     DECW    24(R0)
08        6A       18 E5 00038 2$: BBCC    #24, (BASE), 3$                               ; 1699
                0C AA DD 0003C     PUSHL   12(BASE)                                      ; 1700
    0000G CF       01 FB 0003F     CALLS   #1, CLOSE_FILE
05        6A       13 E5 00044 3$: BBCC    #19, (BASE), 4$                               ; 1705
    0000V CF       00 FB 00048     CALLS   #0, FLUSH_FIDCACHE                            ; 1706
05        6A       19 E5 0004D 4$: BBCC    #25, (BASE), 5$                               ; 1712
    0000G CF       00 FB 00051     CALLS   #0, DEACC_QFILE                               ; 1713
                04 AA D5 00056 5$: TSTL    4(BASE)                                       ; 1718
                   08 13 00059     BEQL    6$
                04 AA DD 0005B     PUSHL   4(BASE)                                       ; 1719
    0000G CF       01 FB 0005E     CALLS   #1, CHECKSUM
05        6A       11 E5 00063 6$: BBCC    #17, (BASE), 7$                               ; 1724
    0000V CF       00 FB 00067     CALLS   #0, ZERO_CHANNEL                              ; 1725
          52    28 AA D0 0006C 7$: MOVL    40(BASE), UNREC_LOCAL                         ; 1730
                   17 13 00070     BEQL    8$
                28 AA D4 00072     CLRL    40(BASE)                                      ; 1733
                2C AA DD 00075     PUSHL   44(BASE)                                      ; 1734
    0000G CF       01 FB 00078     CALLS   #1, SWITCH_VOLUME
                   7E D4 0007D     CLRL    -(SP)                                         ; 1735
                   52 DD 0007F     PUSHL   UNREC_LOCAL
                24 AA DD 00081     PUSHL   36(BASE)
    0000G CF       03 FB 00084     CALLS   #3, RETURN_BLOCKS
          52    A8 AA D0 00089 8$: MOVL    -88(BASE), FID_LOCAL                          ; 1742
                   12 13 0008D     BEQL    9$
                A8 AA D4 0008F     CLRL    -88(BASE)                                     ; 1745
                AC AA DD 00092     PUSHL   -84(BASE)                                     ; 1746
    0000G CF       01 FB 00095     CALLS   #1, SWITCH_VOLUME
                   52 DD 0009A     PUSHL   FID_LOCAL                                     ; 1747
    0000G CF       01 FB 0009C     CALLS   #1, DELETE_FID
                   56 D4 000A1 9$: CLRL    HEADER                                        ; 1753
                04 AA D5 000A3     TSTL    4(BASE)                                       ; 1754
                   23 13 000A6     BEQL    12$
```

```
                                   04    AA  D4 000A8         CLRL    4(BASE)                                   ; 1757
                     14    AA      18    AA  D0 000AB         MOVL    24(BASE), 20(BASE)                        ; 1758
                                         19  13 000B0         BEQL    12$                                       ;
                                   00    BE  DD 000B2         PUSHL   @0(SP)                                    ; 1763
                                         69  D5 000B5         TSTL    (R9)                                      ; 1760
                                         08  13 000B7         BEQL    10$                                       ;
           50              69            04  C1 000B9         ADDL3   #4, (R9), R0                              ; 1761
                                         50  DD 000BD         PUSHL   R0                                        ;
                                         02  11 000BF         BRB     11$                                       ;
                                         7E  D4 000C1  10$:   CLRL    -(SP)                                     ;
                  0000G    CF            02  FB 000C3  11$:   CALLS   #2, READ_HEADER                           ; 1760
                           56            50  D0 000C8         MOVL    R0, HEADER                                ;
           11              6A            02  E5 000CB  12$:   BBCC    #2, (BASE), 13$                           ; 1769
                  0000G    CF            00  FB 000CF         CALLS   #0, ALLOCATION_UNLOCK                     ; 1777
                                   00    BE  DD 000D4         PUSHL   @0(SP)                                    ; 1778
                                         56  DD 000D7         PUSHL   HEADER                                    ;
              00000000G    00            02  FB 000D9         CALLS   #2, SEND_SYMBIONT                         ;
           05              6A            10  E5 000E0  13$:   BBCC    #16, (BASE), 14$                          ; 1784
                  0000V    CF            00  FB 000E4         CALLS   #0, MAKE_DEACCESS                         ; 1785
           1C              6A            1A  E5 000E9  14$:   BBCC    #26, (BASE), 16$                          ; 1790
                                   0C    AA  D5 000ED         TSTL    12(BASE)                                  ; 1792
                                         17  13 000F0         BEQL    16$                                       ;
                           52      0C    AA  D0 000F2         MOVL    12(BASE), WINDOW_SEGMENT                  ; 1795
                           53      20    A2  D0 000F6  15$:   MOVL    32(WINDOW_SEGMENT), NEXT_SEGMENT          ; 1798
                                         52  DD 000FA         PUSHL   WINDOW_SEGMENT                            ; 1799
                  0000G    CF            01  FB 000FC         CALLS   #1, DEALLOCATE                            ;
                           52            53  D0 00101         MOVL    NEXT_SEGMENT, WINDOW_SEGMENT              ; 1800
                                         F0  12 00104         BNEQ    15$                                       ; 1802
                                   0C    AA  D4 00106         CLRL    12(BASE)                                  ; 1803
           29              6A            1E  E5 00109  16$:   BBCC    #30, (BASE), 17$                          ; 1809
                                         56  D5 0010D         TSTL    HEADER                                    ; 1810
                                         25  13 0010F         BEQL    17$                                       ;
      42   A6      30      AA            06  28 00111         MOVC3   #6, 48(BASE), 66(HEADER)                  ; 1813
                           50            66  9A 00117         MOVZBL  (HEADER), R0                              ; 1814
                           58          6640  3E 0011A         MOVAW   (HEADER)[R0], IDENT_AREA                  ;
           68              6B            14  28 0011E         MOVC3   #20, (R11), (IDENT_AREA)                  ; 1816
      36   A8      14      AB            3C  28 00122         MOVC3   #60, 20(R11), 54(IDENT_AREA)              ; 1819
                                         56  DD 00128         PUSHL   HEADER                                    ; 1820
                  0000G    CF            01  FB 0012A         CALLS   #1, CHECKSUM                              ;
                                         56  DD 0012F         PUSHL   HEADER                                    ; 1821
                  0000G    CF            01  FB 00131         CALLS   #1, MARK_DIRTY                            ;
           24              6A            15  E5 00136  17$:   BBCC    #21, (BASE), 19$                          ; 1828
                                         56  D5 0013A         TSTL    HEADER                                    ; 1829
                                         20  13 0013C         BEQL    19$                                       ;
                           50      00    BE  D0 0013E         MOVL    @0(SP), R0                                ; 1832
                                         0D  13 00142         BEQL    18$                                       ;
                                 00B0    C0  D5 00144         TSTL    176(R0)                                   ; 1834
                                         07  13 00148         BEQL    18$                                       ;
                                         50  DD 0014A         PUSHL   R0                                        ; 1836
                  0000G    CF            01  FB 0014C         CALLS   #1, KILL_DINDX                            ;
                     02    AA            04  8A 00151  18$:   BICB2   #4, 2(BASE)                               ; 1838
                                         56  DD 00155         PUSHL   HEADER                                    ; 1839
                                         69  DD 00157         PUSHL   (R9)                                      ;
                  0000G    CF            02  FB 00159         CALLS   #2, DELETE_FILE                           ;
           4F              6A            12  E5 0015E  19$:   BBCC    #18, (BASE), 20$                          ; 1845
                                         56  D5 00162         TSTL    HEADER                                    ; 1846
                                         4B  13 00164         BEQL    20$                                       ;
```

```
CLENUP                                                      F 13
V04-000                                                     16-Sep-1984 00:02:25   VAX-11 Bliss-32 V4.0-742                 Page  25
                                                            14-Sep-1984 12:30:12   DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1     (5)

                        50                69 D0 00166       MOVL    (R9), R0                            ; 1849
                        54          18    A0 D0 00169       MOVL    24(R0), T1
                        50                69 D0 0016D       MOVL    (R9), R0                            ; 1850
                        52          1C    A0 D0 00170       MOVL    28(R0), T2
          50      04    AE                04 C1 00174       ADDL3   #4, 4(SP), R0                       ; 1851
                        53                60 D0 00179       MOVL    (R0), T3
                        50                69 D0 0017C       MOVL    (R9), R0                            ; 1852
                                    18    A0 D4 0017F       CLRL    24(R0)
                                          52 DD 00182       PUSHL   T2                                  ; 1853
                                          56 DD 00184       PUSHL   HEADER
                                          69 DD 00186       PUSHL   (R9)
               0000G    CF                03 FB 00188       CALLS   #3, TRUNCATE
                        56          04    AA D0 0018D       MOVL    4(BASE), HEADER                     ; 1854
                        50                69 D0 00191       MOVL    (R9), R0                            ; 1855
                  18    A0                54 D0 00194       MOVL    T1, 24(R0)
                        50                69 D0 00198       MOVL    (R9), R0                            ; 1856
                  1C    A0                52 D0 0019B       MOVL    T2, 28(R0)
          50      04    AE                04 C1 0019F       ADDL3   #4, 4(SP), R0                       ; 1857
                        60                53 D0 001A4       MOVL    T3, (R0)
                        6A                10 8A 001A7       BICB2   #16, (BASE)                         ; 1858
                                          56 DD 001AA       PUSHL   HEADER                              ; 1859
               0000G    CF                01 FB 001AC       CALLS   #1, CHECKSUM
          15            6A                01 E5 001B1 20$:  BBCC    #1, (BASE), 21$                     ; 1866
                                          56 D5 001B5       TSTL    HEADER                              ; 1867
                                          11 13 001B7       BEQL    21$
                                          56 DD 001B9       PUSHL   HEADER                              ; 1871
                                    04    BE DD 001BB       PUSHL   @4(SP)
               0000G    CF                02 FB 001BE       CALLS   #2, REBLD_PRIM_FCB
                                          56 DD 001C3       PUSHL   HEADER                              ; 1873
               0000G    CF                01 FB 001C5       CALLS   #1, BUILD_EXT_FCBS
          05            6A                1F E5 001CA 21$:  BBCC    #31, (BASE), 22$                    ; 1880
               0000G    CF                00 FB 001CE       CALLS   #0, REMAP_FILE
                        58                6A D0 001D3 22$:  MOVL    (BASE), DIR_FLAGS                   ; 1887
                        6A    00C00020    8F CA 001D6       BICL2   #12582944, (BASE)                   ; 1890
          0B            58                05 E0 001DD       BBS     #5, DIR_FLAGS, 23$                  ; 1892
          07            58                17 E0 001E1       BBS     #23, DIR_FLAGS, 23$                 ; 1893
          03            58                16 E0 001E5       BBS     #22, DIR_FLAGS, 23$                 ; 1894
                                        009E 31 001E9       BRW     28$
                        50                69 D0 001EC 23$:  MOVL    (R9), R0                            ; 1897
                        7E          0E    A0 3C 001EF       MOVZWL  14(R0), -(SP)
               0000G    CF                01 FB 001F3       CALLS   #1, SWITCH_VOLUME
                                    08    A7 D5 001F8       TSTL    8(R7)                               ; 1903
                                          07 13 001FB       BEQL    24$
                                          57 DD 001FD       PUSHL   R7                                  ; 1904
               0000G    CF                01 FB 001FF       CALLS   #1, RESTORE_DIR
          07            58                16 E1 00204 24$:  BBC     #22, DIR_FLAGS, 25$                 ; 1910
                                          7E D4 00208       CLRL    -(SP)                               ; 1911
               0000G    CF                01 FB 0020A       CALLS   #1, REMOVE
          52            58                17 E1 0020F 25$:  BBC     #23, DIR_FLAGS, 27$                 ; 1920
  10      00            6E                00 2C 00213       MOVC5   #0, (SP), #0, #16, NAME_DESC        ; 1923
                                    08    AE    00218
                  0C    AE        0156    CA 9A 0021A       MOVZBL  342(BASE), NAME_DESC+4              ; 1924
                  10    AE        0157    CA 9E 00220       MOVAB   343(BASE), NAME_DESC+8              ; 1925
                  14    AE        0152    CA B0 00226       MOVW    338(BASE), NAME_DESC+12             ; 1926
          1E            58                05 E1 0022C       BBC     #5, DIR_FLAGS, 26$                  ; 1927
                                    1C    A7 94 00230       CLRB    28(R7)                              ; 1930
                        7E                01 CE 00233       MNEGL   #1, -(SP)                           ; 1931
```

```
CLENUP                                            G 13
V04-000                                           16-Sep-1984 00:02:25    VAX-11 Bliss-32 V4.0-742                   Page  26
                                                  14-Sep-1984 12:30:12    DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1      (5)

                                  7E  7C 00236          CLRQ    -(SP)                                        ;
                                  7E  7C 00238          CLRQ    -(SP)                                        ;
                                  7E  D4 0023A          CLRL    -(SP)                                        ;
                          20      AE  9F 0023C          PUSHAB  NAME_DESC                                    ;
              0000G  CF           07  FB 0023F          CALLS   #7, DIR_SCAN                                 ;
                     50           69  D0 00244          MOVL    (R9), R0                                     ; 1932
   04    A0   01FE   CA           06  28 00247          MOVC3   #6, 510(BASE), 4(R0)                         ;
                                  69  DD 0024E 26$:     PUSHL   (R9)                                         ; 1934
                          0C      AE  9F 00250          PUSHAB  NAME_DESC                                    ;
              0000G  CF           02  FB 00253          CALLS   #2, MAKE_ENTRY                               ;
                 02  AA       40  8F  8A 00258          BICB2   #64, 2(BASE)                                 ; 1935
                          04      A7  DD 0025D          PUSHL   4(R7)                                        ; 1936
              0000G  CF           01  FB 00260          CALLS   #1, WRITE_BLOCK                              ;
         21          58           05  E1 00265 27$:     BBC     #5, DIR_FLAGS, 28$                           ; 1945
         1D          58           17  E0 00269          BBS     #23, DIR_FLAGS, 28$                          ; 1946
         19          58           16  E0 0026D          BBS     #22, DIR_FLAGS, 28$                          ; 1947
                0C   B7     0152  CA  B0 00271          MOVW    338(BASE), @12(R7)                           ; 1950
                     50       0C  A7  D0 00277          MOVL    12(R7), R0                                   ; 1951
   02    A0   01FE   CA           06  28 0027B          MOVC3   #6, 510(BASE), 2(R0)                         ;
                          04      A7  DD 00282          PUSHL   4(R7)                                        ; 1952
              0000G  CF           01  FB 00285          CALLS   #1, MARK_DIRTY                               ;
                          36      AA  D5 0028A 28$:     TSTL    54(BASE)                                     ; 1962
                                  0B  13 0028D          BEQL    29$                                          ;
         6A     36   AA           36  28 0028F          MOVC3   #54, 54(BASE), (BASE)                        ; 1963
                          36      AA  D4 00294          CLRL    54(BASE)                                     ; 1964
                                FD8F  31 00297          BRW     1$                                           ; 1661
                     50           01  D0 0029A 29$:     MOVL    #1, R0                                       ; 1968
                                      04 0029D          RET                                                  ; 1970

; Routine Size:  670 bytes,    Routine Base:  $CODE$ + 0194
```

```
                                                                    H 13
CLENUP                                                              16-Sep-1984 00:02:25     VAX-11 Bliss-32 V4.0-742                 Page  27
V04-000                                                             14-Sep-1984 12:30:12     DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1     (6)

;   986          1971   1 ROUTINE FLUSH_FIDCACHE : L_NORM =
;   987          1972   1
;   988          1973   1 !++
;   989          1974   1 !
;   990          1975   1 ! FUNCTIONAL DESCRIPTION:
;   991          1976   1 !
;   992          1977   1 !       This routine empties the file ID cache by zeroing the entry count.
;   993          1978   1 !       It must be called in kernel mode.
;   994          1979   1 !
;   995          1980   1 !
;   996          1981   1 ! CALLING SEQUENCE:
;   997          1982   1 !       FLUSH_FIDCACHE ()
;   998          1983   1 !
;   999          1984   1 ! INPUT PARAMETERS:
;  1000          1985   1 !       NONE
;  1001          1986   1 !
;  1002          1987   1 ! IMPLICIT INPUTS:
;  1003          1988   1 !       CURRENT_VCB: VCB of volume
;  1004          1989   1 !
;  1005          1990   1 ! OUTPUT PARAMETERS:
;  1006          1991   1 !       NONE
;  1007          1992   1 !
;  1008          1993   1 ! IMPLICIT OUTPUTS:
;  1009          1994   1 !       NONE
;  1010          1995   1 !
;  1011          1996   1 ! ROUTINE VALUE:
;  1012          1997   1 !       1
;  1013          1998   1 !
;  1014          1999   1 ! SIDE EFFECTS:
;  1015          2000   1 !       file ID cache cleared
;  1016          2001   1 !
;  1017          2002   1 !--
;  1018          2003   1
;  1019          2004   2 BEGIN
;  1020          2005   2
;  1021          2006   2 BIND_COMMON;
;  1022          2007   2
;  1023          2008   2 LOCAL
;  1024          2009   2       FID_CACHE       : REF BBLOCK;   ! file ID cache
;  1025          2010   2
;  1026          2011   2
;  1027          2012   2 FID_CACHE = .BBLOCK [.CURRENT_VCB[VCB$L_CACHE], VCA$L_FIDCACHE];
;  1028          2013   2 FID_CACHE[VCA$W_FIDCOUNT] = 0;
;  1029          2014   2
;  1030          2015   2 1
;  1031          2016   1 END;                                    ! end of routine FLUSH_FIDCACHE


                                                  0000 00000 FLUSH_FIDCACHE:
                                                                   .WORD    Save nothing                                  ; 1971
                              50      98    AA    D0 00002         MOVL     -104(BASE), R0                                ; 2012
                              50      58    B0    D0 00006         MOVL     @88(R0), FID_CACHE                            ;
                                      02    A0    B4 0000A         CLRW     2(FID_CACHE)                                  ; 2013
                              50            01    D0 0000D         MOVL     #1, R0                                        ; 2016
```

```
                                                                  04 00010          RET                                                   ;

; Routine Size:  17 bytes,    Routine Base:  $CODE$ + 0432
```

```
1 ROUTINE MAKE_DEACCESS : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine performs the machinery for deaccessing a file.
1 !
1 ! CALLING SEQUENCE:
1 !       MAKE_DEACCESS ()
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       PRIMARY_FCB: FCB of file
1 !       CURRENT_WINDOW: window of file
1 !       CURRENT_VCB: VCB of volume in process
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       file deaccessed
1 !
1 !--
1
2 BEGIN
2
2 BIND_COMMON;
2
2 LOCAL
2       FCB             : REF BBLOCK,   ! local for primary fcb.
2       LCKMODE,                        ! lock mode for access lock.
2       WINDOW_SEGMENT  : REF BBLOCK,   ! address of the next window segment
2       DUMMY;                          ! dummy local to receive REMQUE
2
2 EXTERNAL
2       PMS$GL_OPEN     : ADDRESSING_MODE (ABSOLUTE);
2                                       ! system count of currently open files
2
2 EXTERNAL ROUTINE
2       DEQ_LOCK        : L_NORM,       ! dequeue a lock
2       CONV_ACCLOCK    : L_NORM,       ! Convert file access lock.
2       LOCK_MODE       : L_JSB_1ARG;   ! Calculate access lock mode.
2
2 FCB = .PRIMARY_FCB;
2
2 ! Unlink the window from the FCB. Clear the applicable access conditions
2 ! in the FCB.
2 !
```

```
2
2 WINDOW_SEGMENT = .CURRENT_WINDOW;
2 DO
3     BEGIN
3     IF .WINDOW_SEGMENT[WCB$L_WLFL] NEQ 0 THEN REMQUE (.WINDOW_SEGMENT, DUMMY);
3     WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
3     END
2 UNTIL .WINDOW_SEGMENT EQL 0;
2
2 IF NOT .CURRENT_WINDOW [WCB$V_NOACCLOCK]
2 THEN
3     BEGIN
3     IF .CURRENT_WINDOW[WCB$V_NOREAD]
3     THEN FCB[FCB$V_EXCL] = 0;
3
3     IF .CURRENT_WINDOW[WCB$V_NOTRUNC]
3     THEN FCB[FCB$W_TCNT] = .FCB[FCB$W_TCNT] - 1;
3
3     IF .CURRENT_WINDOW[WCB$V_NOWRITE]
3     THEN FCB[FCB$W_LCNT] = .FCB[FCB$W_LCNT] - 1;
3
3     FCB [FCB$W_ACNT] = .FCB [FCB$W_ACNT] - 1;
3
2     END;                              ! of normal (not NOLOCK) deaccess.
2
2 FCB[FCB$W_REFCNT] = .FCB[FCB$W_REFCNT] - 1;
2
2 ! For a write access, bump down the writer count. If this is the
2 ! last write, and the file is the index file or the storage map, clear
2 ! the appropriate flag in the VCB. If there's a cache lock being held
2 ! for this file, release it.
2 !
2
2 IF .CURRENT_WINDOW[WCB$V_WRITE]
2 THEN
3     BEGIN
3
3     IF NOT .CURRENT_WINDOW [WCB$V_NOACCLOCK]
3     THEN
3         FCB[FCB$W_WCNT] = .FCB[FCB$W_WCNT] - 1;
3
3     IF .FCB[FCB$W_WCNT] EQL 0
4         OR (.FCB [FCB$W_REFCNT] EQL 0 AND .CURRENT_WINDOW [WCB$V_WRITE])
3     THEN
4         BEGIN
4         IF .FCB[FCB$B_FID_NMX] EQL 0
4         THEN
5             BEGIN
5             IF .FCB[FCB$W_FID_NUM] EQL 1
5             THEN CURRENT_VCB[VCB$V_WRITE_IF] = 0;
5             IF .FCB[FCB$W_FID_NUM] EQL 2
5             THEN CURRENT_VCB[VCB$V_WRITE_SM] = 0;
5             END;
4         IF .FCB[FCB$L_CACHELKID] NEQ 0
4         THEN
5             BEGIN
5             DEQ_LOCK (.FCB[FCB$L_CACHELKID]);
```

```
5                       FCB[FCB$L_CACHELKID] = 0;
4                       END;
3               END;
2           END;
2
2   ! Recalculate the lock mode of the access lock for this fcb.
2   !
2
2   IF .FCB [FCB$W_ACNT] EQL 0
2   THEN
2       LCKMODE = LCK$K_NLMODE
2   ELSE
3       BEGIN
3       LOCAL
3           ACCTL;
3
3       ACCTL = 0;
3       IF .FCB [FCB$W_WCNT] NEQ 0
3       THEN ACCTL = .ACCTL + FIB$M_WRITE;
3       IF .FCB [FCB$W_LCNT] NEQ 0
3       THEN ACCTL = .ACCTL + FIB$M_NOWRITE;
3
3       LCKMODE = LOCK_MODE (.ACCTL);
3
2       END;
2
2   ! If the new access lock mode lock for this fcb is different (lower)
2   ! than the current lock, convert it.  The conversion routine will also
2   ! dequeue the lock if this is the last reference.
2   !
2
2   IF .LCKMODE<0,8> NEQ .FCB [FCB$B_ACCLKMODE]
2       OR .FCB [FCB$W_REFCNT] EQL 0
2   THEN
2       IF NOT CONV_ACCLOCK (.LCKMODE, .FCB)
2       THEN
2           BUG_CHECK (XQPERR, 'deaccess conversion failed');
2
2   ! Note: We now have a file control block with a possible zero access count
2   ! in the FCB list. This gets dealt with by the general cleanup.
2   !
2
2   PMS$GL_OPEN = .PMS$GL_OPEN - 1;         ! bump down count of open files
2   CURRENT_VCB[VCB$W_TRANS] = .CURRENT_VCB[VCB$W_TRANS] - 1;
2
2   RETURN 1;
2
1   END;                                    ! end of routine MAKE_DEACCESS


                                                    .EXTRN  PMS$GL_OPEN, DEQ_LOCK
                                                    .EXTRN  CONV_ACCLOCK, LOCK_MODE
                                                    .EXTRN  BUG$_XQPERR

                                  000C 00000 MAKE_DEACCESS:
                                                    .WORD   Save R2,R3                          ; 2017
```

```
              51      0C    AA   9E 00002        MOVAB    12(BASE), R1                                  ; 2050
              52      08    AA   D0 00006        MOVL     8(BASE), FCB                                  ; 2069
              50            61   D0 0000A        MOVL     (R1), WINDOW_SEGMENT                          ; 2075
                            60   D5 0000D 1$:    TSTL     (WINDOW_SEGMENT)                              ; 2078
                            03   13 0000F        BEQL     2$                                            ;
              53            60   0F 00011        REMQUE   (WINDOW_SEGMENT), DUMMY                       ;
              50      20    A0   D0 00014 2$:    MOVL     32(WINDOW_SEGMENT), WINDOW_SEGMENT            ; 2079
                            F3   12 00018        BNEQ     1$                                            ; 2081
              50            61   D0 0001A        MOVL     (R1), R0                                      ; 2083
21      14    A0            02   E0 0001D        BBS      #2, 20(R0), 6$                                ;
04      15    A0            02   E1 00022        BBC      #2, 21(R0), 3$                                ; 2086
        22    A2            08   8A 00027        BICB2    #8, 34(FCB)                                   ; 2087
              50            61   D0 0002B 3$:    MOVL     (R1), R0                                      ; 2089
03      15    A0            03   E1 0002E        BBC      #3, 21(R0), 4$                                ;
                      20    A2   B7 00033        DECW     32(FCB)                                       ; 2090
              50            61   D0 00036 4$:    MOVL     (R1), R0                                      ; 2092
              03      14    A0   E9 00039        BLBC     20(R0), 5$                                    ;
                      1E    A2   B7 0003D        DECW     30(FCB)                                       ; 2093
                      1A    A2   B7 00040 5$:    DECW     26(FCB)                                       ; 2095
                      18    A2   B7 00043 6$:    DECW     24(FCB)                                       ; 2099
              50            61   D0 00046        MOVL     (R1), R0                                      ; 2107
4B      0B    A0            01   E1 00049        BBC      #1, 11(R0), 11$                               ;
03      14    A0            02   E0 0004E        BBS      #2, 20(R0), 7$                                ; 2111
                      1C    A2   B7 00053        DECW     28(FCB)                                       ; 2113
                      1C    A2   B5 00056 7$:    TSTW     28(FCB)                                       ; 2115
                            0D   13 00059        BEQL     8$                                            ;
                      18    A2   B5 0005B        TSTW     24(FCB)                                       ; 2116
                            39   12 0005E        BNEQ     11$                                           ;
              50            61   D0 00060        MOVL     (R1), R0                                      ;
31      0B    A0            01   E1 00063        BBC      #1, 11(R0), 11$                               ;
                      29    A2   95 00068 8$:    TSTB     41(FCB)                                       ; 2119
                            1C   12 0006B        BNEQ     10$                                           ;
              01      24    A2   B1 0006D        CMPW     36(FCB), #1                                   ; 2122
                            08   12 00071        BNEQ     9$                                            ;
              50      98    AA   D0 00073        MOVL     -104(BASE), R0                                ; 2123
        0B    A0            01   8A 00077        BICB2    #1, 11(R0)                                    ;
              02      24    A2   B1 0007B 9$:    CMPW     36(FCB), #2                                   ; 2124
                            08   12 0007F        BNEQ     10$                                           ;
              50      98    AA   D0 00081        MOVL     -104(BASE), R0                                ; 2125
        0B    A0            02   8A 00085        BICB2    #2, 11(R0)                                    ;
                      54    A2   D5 00089 10$:   TSTL     84(FCB)                                       ; 2127
                            0B   13 0008C        BEQL     11$                                           ;
                      54    A2   DD 0008E        PUSHL    84(FCB)                                       ; 2130
     0000G    CF            01   FB 00091        CALLS    #1, DEQ_LOCK                                  ;
                      54    A2   D4 00096        CLRL     84(FCB)                                       ; 2131
                      1A    A2   B5 00099 11$:   TSTW     26(FCB)                                       ; 2139
                            04   12 0009C        BNEQ     12$                                           ;
                            51   D4 0009E        CLRL     LCKMODE                                       ; 2141
                            19   11 000A0        BRB      15$                                           ;
                            50   D4 000A2 12$:   CLRL     ACCTL                                         ; 2147
                      1C    A2   B5 000A4        TSTW     28(FCB)                                       ; 2148
                            05   13 000A7        BEQL     13$                                           ;
              50    0100    C0   9E 000A9        MOVAB    256(R0), ACCTL                                ; 2149
                      1E    A2   B5 000AE 13$:   TSTW     30(FCB)                                       ; 2150
                            02   13 000B1        BEQL     14$                                           ;
                            50   D6 000B3        INCL     ACCTL                                         ; 2151
                         0000G   30 000B5 14$:   BSBW     LOCK_MODE                                     ; 2153
```

```
                51            50  D0 000B8         MOVL    R0, LCKMODE                              ;
         0B     A2            51  91 000BB 15$:    CMPB    LCKMODE, 11(FCB)                         ; 2162
                              05  12 000BF         BNEQ    16$                                      ;
                      18      A2  B5 000C1         TSTW    24(FCB)                                  ; 2163
                              0E  12 000C4         BNEQ    17$                                      ;
                              06  BB 000C6 16$:    PUSHR   #^M<R1,R2>                               ; 2165
      0000G     CF            02  FB 000C8         CALLS   #2, CONV_ACCLOCK                         ;
                04            50  E8 000CD         BLBS    R0, 17$                                  ;
                                FEFF 000D0         BUGW                                             ; 2167
                               0000* 000D2         .WORD   <BUG$_XQPERR!4>                          ;
                   00000000G  9F  D7 000D4 17$:    DECL    @#PMS$GL_OPEN                            ; 2173
                50         98 AA  D0 000DA         MOVL    -104(BASE), R0                           ; 2174
                           0C A0  B7 000DE         DECW    12(R0)                                   ;
                50            01  D0 000E1         MOVL    #1, R0                                   ; 2176
                                  04 000E4         RET                                              ; 2178

; Routine Size:  229 bytes,    Routine Base:  $CODE$ + 0443
```

```
1 GLOBAL ROUTINE DEL_EXTFCB (START_FCB) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine removes and deallocates all extension FCB's, if any,
1 !       linked to the indicated FCB.
1 !
1 ! CALLING SEQUENCE:
1 !       DEL_EXTFCB (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of primary FCB or 0
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       FCB's deallocated
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       START_FCB       : REF BBLOCK;   ! FCB argument
2
2 LOCAL
2       FCB             : REF BBLOCK,   ! running FCB pointer
2       NEXT_FCB        : REF BBLOCK,   ! next extension FCB
2       P               : REF BBLOCK,   ! pointer to chase for VCB
2       DUMMY;                          ! dummy local to receive REMQUE
2
2 BASE_REGISTER;
2
2 EXTERNAL ROUTINE
2       DEALLOCATE      : L_NORM;       ! deallocate dynamic memory
2
2 ! Checking for null pointers, find the first extension FCB. Follow the extension
2 ! list and remove and deallocate the extension FCB's, cleaning out the pointers
2 ! on the way. For each FCB removed, we must find the VCB (by chasing around the
2 ! FCB list) and decrement the transaction count.
2 !
2
2 IF .START_FCB EQL 0 THEN RETURN 1;
2 FCB = .START_FCB[FCB$L_EXFCB];
2 START_FCB[FCB$L_EXFCB] = 0;
```

```
; 1253          2236  2 UNTIL .FCB EQL 0 DO
; 1254          2237  3      BEGIN
; 1255          2238  3      NEXT_FCB = .FCB[FCB$L_EXFCB];
; 1256          2239  3
; 1257          2240  3      P = .FCB[FCB$L_FCBFL];
; 1258          2241  3      UNTIL .P[VCB$B_TYPE] EQL DYN$C_VCB
; 1259          2242  3      DO P = .P[FCB$L_FCBFL];
; 1260          2243  3      P[VCB$W_TRANS] = .P[VCB$W_TRANS] - 1;
; 1261          2244  3
; 1262          2245  3      FCB[FCB$L_EXFCB] = 0;
; 1263          2246  3      IF .FCB [FCB$B_TYPE] NEQ DYN$C_FCB
; 1264          2247  3      THEN
; 1265          2248  3          BUG_CHECK (NOTFCBFCB, 'not fcb');
; 1266          2249  3      REMQUE (.FCB, DUMMY);
; 1267          2250  3      DEALLOCATE (.FCB);
; 1268          2251  3      FCB = .NEXT_FCB;
; 1269          2252  2      END;
; 1270          2253  2
; 1271          2254  2 RETURN 1;
; 1272          2255  2
; 1273          2256  1 END;                                         ! end of routine DEL_EXTFCB


                                                                           .EXTRN  BUGS_NOTFCBFCB

                                                 003C 00000                .ENTRY  DEL_EXTFCB, Save R2,R3,R4,R5                     ; 2179
                             50        04    AC    D0 00002                MOVL    START_FCB, R0                                    ; 2233
                                             3C    13 00006                BEQL    5$                                               ;
                             53        0C    A0    D0 00008                MOVL    12(R0), FCB                                      ; 2234
                                       0C    A0    D4 0000C                CLRL    12(R0)                                           ; 2235
                                             53    D5 0000F  1$:           TSTL    FCB                                              ; 2236
                                             31    13 00011                BEQL    5$                                               ;
                             54        0C    A3    D0 00013                MOVL    12(FCB), NEXT_FCB                                ; 2238
                             52              63    D0 00017                MOVL    (FCB), P                                         ; 2240
                             11        0A    A2    91 0001A  2$:           CMPB    10(P), #17                                       ; 2241
                                             05    13 0001E                BEQL    3$                                               ;
                             52              62    D0 00020                MOVL    (P), P                                           ; 2242
                                             F5    11 00023                BRB     2$                                               ;
                                       0C    A2    B7 00025  3$:           DECW    12(P)                                            ; 2243
                                       0C    A3    D4 00028                CLRL    12(FCB)                                          ; 2245
                             07        0A    A3    91 0002B                CMPB    10(FCB), #7                                      ; 2246
                                             04    13 0002F                BEQL    4$                                               ;
                                                FEFF 00031                 BUGW                                                     ; 2248
                                               0000* 00033                 .WORD   <BUG$_NOTFCBFCB!4>                               ;
                             55              63    0F 00035  4$:           REMQUE  (FCB), DUMMY                                     ; 2249
                                             53    DD 00038                PUSHL   FCB                                              ; 2250
                      0000G  CF              01    FB 0003A                CALLS   #1, DEALLOCATE                                   ;
                             53              54    D0 0003F                MOVL    NEXT_FCB, FCB                                    ; 2251
                                             CB    11 00042                BRB     1$                                               ; 2236
                             50              01    D0 00044  5$:           MOVL    #1, R0                                           ; 2254
                                                   04 00047                RET                                                      ; 2256

; Routine Size:  72 bytes,    Routine Base:  $CODE$ + 0528
```

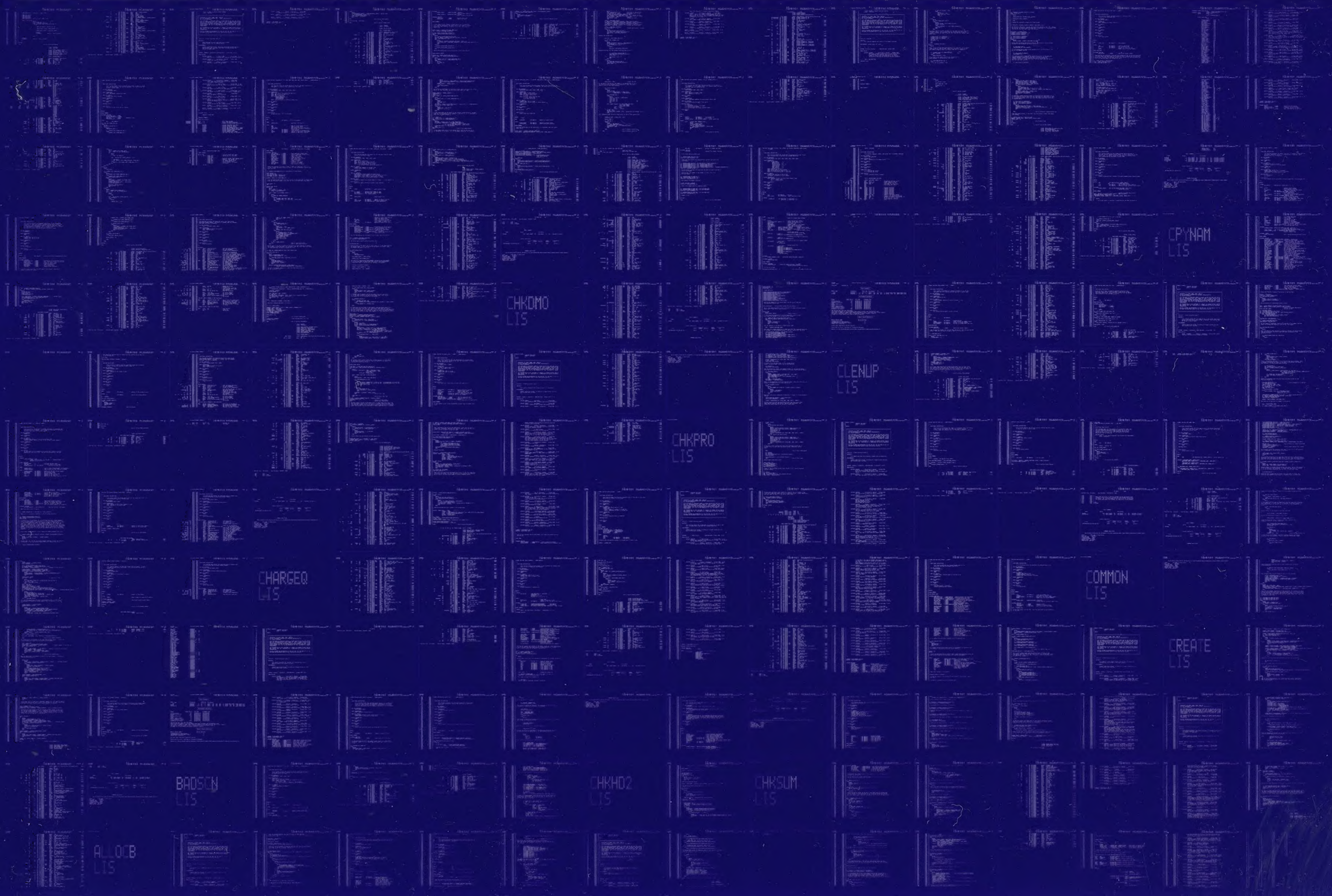
```
ROUTINE ZERO_CHANNEL : L_NORM =

!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine zeroes out the window pointer being returned to
!       the user for his channel control block. It also credits one to the
!       user's open file quota, except for the case of a shared window.
!       This routine must be executed in kernel mode.
!
! CALLING SEQUENCE:
!       ZERO_CHANNEL ()
!
! INPUT PARAMETERS:
!       NONE
!
! IMPLICIT INPUTS:
!       IO_PACKET: I/O packet of request
!
! OUTPUT PARAMETERS:
!       NONE
!
! IMPLICIT OUTPUTS:
!       NONE
!
! ROUTINE VALUE:
!       NONE
!
! SIDE EFFECTS:
!       channel window pointer cleared, file quota bumped unless shared window
!
!--

BEGIN

LOCAL
        ABD             : REF BBLOCKVECTOR [,ABD$C_LENGTH],
                                        ! buffer descriptors
        JIB             : REF BBLOCK,   ! Job information block address
        PCB             : REF BBLOCK;   ! address of user process control block

EXTERNAL
        SCH$GL_PCBVEC   : REF VECTOR ADDRESSING_MODE (ABSOLUTE);
                                        ! system PCB vector

BIND_COMMON;

                                        ! pointer to buffer descriptors
ABD = .BBLOCK [.IO_PACKET[IRP$L_SVAPTE], AIB$L_DESCRIPT];
ABD[ABD$C_WINDOW, ABD$W_COUNT] = 4;
.ABD[ABD$C_WINDOW, ABD$W_TEXT] + ABD[ABD$C_WINDOW, ABD$W_TEXT] + 1 = 0;

IF
    BEGIN

    ! The FILCNT quota is credited if a WCB has not yet been allocated or
```

```
; 1332          2314  3         ! if the SHRWCB bit is not set in the WCB.
; 1333          2315  3
; 1334          2316  3         IF .CURRENT_WINDOW EQL 0
; 1335          2317  3         THEN 1
; 1336          2318  3         ELSE NOT .CURRENT_WINDOW[WCB$V_SHRWCB]
; 1337          2319  3         END
; 1338          2320  2     THEN
; 1339          2321  3         BEGIN
; 1340          2322  3         PCB = .SCH$GL_PCBVEC[.(IO_PACKET[IRP$L_PID])<0,16>];
; 1341          2323  3         JIB = .PCB[PCB$L_JIB];
; 1342          2324  3         JIB[JIB$W_FILCNT] = .JIB[JIB$W_FILCNT] + 1;
; 1343          2325  2         END;
; 1344          2326  2
; 1345          2327  2     RETURN 1;
; 1346          2328  2
; 1347          2329  1     END;                                  ! end of routine ZERO_CHANNEL


                                                                  .EXTRN  SCH$GL_PCBVEC

                                                  0000 00000 ZERO_CHANNEL:
                                                                  .WORD   Save nothing                                    ; 2257
                          50          90    AA  D0 00002          MOVL    -112(BASE), R0                                  ; 2306
                          51          2C    B0  D0 00006          MOVL    @44(R0), ABD                                    ;
                    02    A1                04  B0 0000A          MOVW    #4, 2(ABD)                                      ; 2307
                          50                61  3C 0000E          MOVZWL  (ABD), R0                                       ; 2308
                                   01 A140  9F 00011              PUSHAB  1(ABD)[R0]                                      ;
                                            9E  D4 00015          CLRL    @(SP)+                                          ;
                          50          0C    AA  D0 00017          MOVL    12(BASE), R0                                    ; 2316
                                            05  13 0001B          BEQL    1$                                              ;
           1D       0B    A0                03  E0 0001D          BBS     #3, 11(R0), 2$                                  ; 2318
                          51   00000000G    9F  D0 00022 1$:      MOVL    @#SCH$GL_PCBVEC, R1                             ; 2322
                          50          90    AA  D0 00029          MOVL    -112(BASE), R0                                  ;
                          50                0C  C0 0002D          ADDL2   #12, R0                                         ;
                          50                60  3C 00030          MOVZWL  (R0), R0                                        ;
                          50              6140  D0 00033          MOVL    (R1)[R0], PCB                                   ;
                          50        0080    C0  D0 00037          MOVL    128(PCB), JIB                                   ; 2323
                                      30    A0  B6 0003C          INCW    48(JIB)                                         ; 2324
                          50                01  D0 0003F 2$:      MOVL    #1, R0                                          ; 2327
                                                04 00042          RET                                                     ; 2329

; Routine Size:  67 bytes,    Routine Base:  $CODE$ + 0570
; 1349          2330  1 GLOBAL ROUTINE NUKE_HEAD_FCB (FCB) : L_NORM NOVALUE =
; 1350          2331  1
; 1351          2332  1 !++
; 1352          2333  1 !
; 1353          2334  1 !  Functional Description:
; 1354          2335  1 !
; 1355          2336  1 !  Given an fcb already stripped of possible extension fcbs,
; 1356          2337  1 !  and which has a refcnt of 0 (assumed), clean up the things
; 1357          2338  1 !  that need cleaning up, remove it from the fcb list (we assume
; 1358          2339  1 !  that is where it is), and deallocate it.
; 1359          2340  1 !
; 1360          2341  1 !--
; 1361          2342  1
; 1362          2343  2 BEGIN
; 1363          2344  2
; 1364          2345  2 MAP
; 1365          2346  2         FCB     : REF BBLOCK;
; 1366          2347  2
; 1367          2348  2 BASE_REGISTER;
; 1368          2349  2
; 1369          2350  2 EXTERNAL ROUTINE
; 1370          2351  2         ACL_DELETEACL,
; 1371          2352  2         CONV_ACCLOCK      : L_NORM,
; 1372          2353  2         DEALLOCATE        : L_NORM;
; 1373          2354  2
; 1374          2355  2 LOCAL
; 1375          2356  2         DUMMY;
; 1376          2357  2
; 1377          2358  2 IF .FCB [FCB$B_TYPE] NEQ DYN$C_FCB
; 1378          2359  2 THEN
; 1379          2360  2     BUG_CHECK (NOTFCBFCB, 'not fcb');
; 1380          2361  2
; 1381          2362  2 REMQUE (.FCB, DUMMY);
; 1382          2363  2
; 1383          2364  2 IF .BBLOCK [FCB [FCB$R_ORB], ORB$V_ACL_QUEUE]
; 1384          2365  2 THEN
; 1385          2366  2     ACL_DELETEACL (FCB [FCB$L_ACLFL], 0);
; 1386          2367  2
; 1387          2368  2 IF NOT CONV_ACCLOCK (0, .FCB)
; 1388          2369  2 THEN
; 1389          2370  2     BUG_CHECK (XQPERR, 'Unexpected lock manager status');
; 1390          2371  2
; 1391          2372  2 DEALLOCATE (.FCB);
; 1392          2373  2
; 1393          2374  1 END;            ! of routine NUKE_HEAD_FCB


                                                                  .EXTRN  ACL_DELETEACL

                                       0000 00000                 .ENTRY  NUKE_HEAD_FCB, Save nothing                      ; 2330
                    50       04   AC   D0 00002                   MOVL    FCB, R0                                          ; 2358
                    07       0A   A0   91 00006                   CMPB    10(R0), #7                                       ;
                                  04   13 0000A                   BEQL    1$                                               ;
                                     FEFF 0000C                   BUGW                                                     ; 2360
                                    0000* 0000E                   .WORD   <BUG$_NOTFCBFCB!4>                               ;
                    50       04   BC   0F 00010 1$:               REMQUE  @FCB, DUMMY                                      ; 2362
```

```
                                                         G 14
CLENUP                                                   16-Sep-1984 00:02:25    VAX-11 Bliss-32 V4.0-742                  Page  39
V04-000                                                  14-Sep-1984 12:30:12    DISK$VMSMASTER:[F11X.SRC]CLENUP.B32;1      (10)

                                 50         04    AC  D0 00014           MOVL    FCB, R0                                    ; 2364
                   10      63    A0               01  E1 00018           BBC     #1, 99(R0), 2$                             ;
                                                  7E  D4 0001D           CLRL    -(SP)                                      ; 2366
                   7E      04    AC 00000080      8F  C1 0001F           ADDL3   #128, FCB, -(SP)                           ;
                        0000G    CF               02  FB 00028           CALLS   #2, ACL_DELETEACL                          ;
                                            04    AC  DD 0002D  2$:      PUSHL   FCB                                        ; 2368
                                                  7E  D4 00030           CLRL    -(SP)                                      ;
                        0000G    CF               02  FB 00032           CALLS   #2, CONV_ACCLOCK                           ;
                                 04               50  E8 00037           BLBS    R0, 3$                                     ;
                                                    FEFF 0003A           BUGW                                               ; 2370
                                                   0000* 0003C           .WORD   <BUG$_XQPERR!4>                            ;
                                            04    AC  DD 0003E  3$:      PUSHL   FCB                                        ; 2372
                        0000G    CF               01  FB 00041           CALLS   #1, DEALLOCATE                             ;
                                                      04 00046           RET                                                ; 2374

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 05B3
; 1395          2375    1 LOCK_CODE;
; 1396          2376    1 GLOBAL ROUTINE SET_DIRINDX (FCB) : L_JSB_1ARG =
; 1397          2377    1
; 1398          2378    1 !++
; 1399          2379    1 !
; 1400          2380    1 ! Functional Description:
; 1401          2381    1 !
; 1402          2382    1 ! This routine tests for the presence of a directory index, and
; 1403          2383    1 ! set the FCB$V_DIR flag accordingly at SCHED ipl, so at to
; 1404          2384    1 ! interlock with the directory index handling routine which
; 1405          2385    1 ! may be trying to toss it out, and the search_fcb routine,
; 1406          2386    1 ! which also runs at sched ipl.
; 1407          2387    1 !
; 1408          2388    1 ! ROUTINE VALUE:
; 1409          2389    1 !       true - if this now a directory fcb eligible for replacement
; 1410          2390    1 !       false - otherwise
; 1411          2391    1 !
; 1412          2392    1 !--
; 1413          2393    1
; 1414          2394    2 BEGIN
; 1415          2395    2
; 1416          2396    2 MAP
; 1417          2397    2         FCB      : REF BBLOCK;
; 1418          2398    2
; 1419          2399    2 LOCAL
; 1420          2400    2         STATUS   : INITIAL (0);
; 1421          2401    2
; 1422          2402    2 SET_IPL (IPL$_SCHED);
; 1423          2403    2
; 1424          2404    2 IF .FCB [FCB$L_DIRINDX] NEQ 0
; 1425          2405    2 THEN
; 1426          2406    3     BEGIN
; 1427          2407    3     FCB [FCB$V_DIR] = 1;
; 1428          2408    3     STATUS = .STATUS + 1;
; 1429          2409    2     END;
; 1430          2410    2
; 1431          2411    2 SET_IPL (0);
; 1432          2412    2
; 1433          2413    2 .STATUS
; 1434          2414    2
; 1435          2415    1 END;             ! of routine SET_DIRINDX



                                                          .PSECT  $LOCKEDC1$,NOWRT,2

                                     51  D4 00000 SET_DIRINDX::
                                                          CLRL    STATUS                              ; 2394
                  12                 03  DA 00002         MTPR    #3, #18                             ; 2402
                          00B0       C0  D5 00005         TSTL    176(FCB)                            ; 2404
                                     06  13 00009         BEQL    1$                                  ;
             22   A0                 01  88 0000B         BISB2   #1, 34(FCB)                         ; 2407
                                     51  D6 0000F         INCL    STATUS                              ; 2408
                  12                 00  DA 00011 1$:     MTPR    #0, #18                             ; 2411
                  50                 51  D0 00014         MOVL    STATUS, R0                          ; 2415
```

```
                                                      05 00017         RSB                                             ;

; Routine Size:  24 bytes,    Routine Base:  $LOCKEDC1$ + 0000


; 1436          2416  1
; 1437          2417  1 ! Note that just prior to the SET_DIRINDX routine the psects were
; 1438          2418  1 ! changed to the locked psect because the SET_DIRINDX routine must
; 1439          2419  1 ! be locked.  Any routines added at this point will be locked also,
; 1440          2420  1 ! so unless they need to be locked, put them prior to SET_DIRINDX.
; 1441          2421  1 !
; 1442          2422  1
; 1443          2423  1 END
; 1444          2424  0 ELUDOM



;                                    PSECT SUMMARY
;
;       Name                       Bytes                        Attributes
;
;  $CODE$                             1530  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $LOCKEDC1$                           24  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                             Library Statistics
;
;                                              -------- Symbols --------      Pages      Processing
;       File                                   Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1               18619        95         0      1000         00:02.0




;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CLENUP/OBJ=OBJ$:CLENUP MSRC$:CLENUP/UPDATE=(ENH$:CLENUP)

; Size:          1554 code + 0 data bytes
; Run Time:         01:19.3
; Elapsed Time:     02:31.2
; Lines/CPU Min:     1834
; Lexemes/CPU-Min: 54610
; Memory Used:  371 pages
; Compilation Complete
```

0168 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY


CPYNAM
LIS

CHKDMO
LIS

CLENUP
LIS

CHKPRO
LIS

CHARGEQ
LIS

COMMON
LIS

CREATE
LIS

BADSCN
LIS

CHKHD2
LIS

CHKSUM
LIS

ALLOCB
LIS